LOUGHBOROUGH UNIVERSITY

Department of Aeronautical and Automotive Engineering

Improving Police Efficiency to meet Demand Issues

O.S.S.T Edleston

December 17, 2010

Abstract

Demand modelling and simulation techniques are used in many industrial practices in order to be able to effectively manage the utilization of available resources. The current economic climate has intensified activity within this field with particular interest being paid to any potential cost savings and other financial benefits that may be obtained. Further the creation of a realistic representation of the demands present within a system can lead to a better understanding of system behaviour; this then may facilitate the identification of elements that are likely to allow improvement to system performance through their perturbation. Within this thesis a model is constructed for the demands upon front line Police officers that are used in response to high importance calls to service from the public.

Tabu search and genetic algorithms are optimizing search techniques developed and applied across a wide variety of fields. They are particularly well suited to combinatorial problems in which the ordering or arrangement of system elements has an impact upon the quality of solution as assessed by some quantifying objective function. In this thesis both of these methods are applied to the staff resource allocation problem as posed by Leicestershire Police with the strengths and weaknesses of each evaluated. Customized diversification and intensification approaches are applied to the tabu search methodology in order to improve performance through tailoring it to the specific optimization problem considered. Both search algorithms are shown to be well suited to the target problem and each result in the generation of solutions of similar quality.

Keywords: Demand Modelling, Tabu Search, Genetic Algorithms, Police Organization.

Acknowledgements

I would like to thank both my supervisor Dr Lisa Jackson and my cover supervisor Dr Sarah Dunnett for their tireless academic support throughout the duration of my research. The assistance and collaboration provided by Leicestershire Police has been invaluable to this research and I would in particular like to thank Mr Dave Hall and Mrs Carolyn Steptoe for their work in supervision and management of the business implementation of my research. My family have provided endless moral support over the past years and as such I would like to thank Mr Steven Edleston, Miss Suzanne Critchell and Mrs Dawn Slade for this. I am also appreciative of the various amongst my friends that have been highly supportive through the duration of my research.

Publications

Conference
Edleston, O. and Bartlett, L.M. Modelling the demands upon UK Police officers and optimization of staffing resources through a tabu search method. Presentation at the OR51 conference, 8-10th September 2009, Warwick University UK.

Conference
Edleston, O. and Bartlett, L.M. An application of tabu search optimization techniques to the problem of Police staff rostering. Presentation at the OR52 conference, 8-10th September 2010, Royal Holloway University UK.

Conference
Edleston, O. and Bartlett, L.M., ‘An Application of Demand Profiling and Optimization of Staffing Levels within Leicestershire Police Force’, Proceedings of the Operational Research Society Simulation Workshop 2010 (SW10), Worcestershire, UK, April 2010, pp210-218.

Journal submission - tracking number JORS-10-0161-P
O.Eldeston and L.Bartlett. ‘A TABU SEARCH ALGORITHM APPLIED TO THE STAFFING ROSTER PROBLEM OF LEICESTERSHIRE Police FORCE’. Submitted to Journal of the Operational Research Society, May 2010. Accepted October 2010.

Contents

List of Figures

List of Tables

Acronyms

ACPO	Association of Chief Police Officers
ARV	Armed Response Vehicle
BCU	Basic Command Unit
CMC	Call Management Centre
FHQ	Force Head Quarters
IACP	International Association of Chiefs of Police
IRV	Incident Response Vehicle
LPO	Local Police Officer
LPU	Local Policing Unit
NIM	National Intelligence Model
OIS	Operational Information System
PBO	Police Beat Officer
PCSO	Police Community Support Officer
RTC	Road Traffic Collision
RTU	Road Traffic Unit

Chapter 1. Introduction to the Thesis

1.1 Introduction

In any country with a developed code of law there is the need to employ some form of policing body to ensure its upkeep and in order to maintain a standard of order and public safety. To this end the United Kingdom nationally employs a total of more than 160000 officers across the entire standard policing structure. This figure is broken down into a variety of necessary officer roles as illustrated in Table 1.1; this is then further spread across all the county based policing regions of the UK.

Table 1.1: UK Officer employment figures for 2006/2007

United Kingdom			Numbers
	Males	Females	All
ACPO Ranks	230	32	262
Chief Superintendent	708	61	769
Superintendent	1056	101	1157
Chief Inspector	2081	261	2341
Inspector	8534	983	8517
Sergeant	21483	3491	24974
Constable	97276	31879	129156
All Ranks	130368	36807	167174

For the county of Leicestershire this results in a total of nearly 2300 officers of all types across the entire region. This figure is further supplemented by approximately 1500 other staff comprising primarily of Call Management Centre (CMC) workers and other desk staff in addition to Police Community Support Officers (PCSOs). Through suitable allocation of these resources Leicestershire Constabulary seeks to provide the best service possible to the public whilst simultaneously maintaining the requirements laid out in the Quality of Service Commitment [1]; this document details 50 key criteria that must be fulfilled by any regional policing body. Additionally a new document entitled the Policing Pledge has been produced by the Police authority and government which enforces further guidelines on Police activity when interacting with members of the public. Thus a major problem that the Police force find themselves confronted with is how to spend the resources they have available in order to best meet all of the demands and constraints made upon the services they provide. This is of particular interest for Leicestershire Police due to a government implemented target of 15 million in revenue savings to be made from April 2011.

This demand comes in a number of forms and originates from a variety of sources, not only do Police officers respond to calls from the general public but they are also constantly undertaking initiatives in order to aid in the prevention and detection of crime. For example major public and sporting events will usually have continued attendance of a number of dedicated officers in addition to generating a sizeable amount of office/paper work. Further there are many tasks that although not directly involved in the detection and resolution of criminal incidents are essential to the smooth and effective running of the force, this includes such positions as superintendent and desk sergeant used in coordinating other active resources. The primary focus of this research is the demand on Local Police Officer (LPO) incident attendance originating from calls made by the general public.

Due to the ever present demand on policing resources it is a viable approach to develop and maintain models in order to promote efficient and purposeful resource allocation. This is the precise goal of the research and, through collaboration with Leicestershire Constabulary, it is intended to further encourage practical applications through trial and implementation of resulting demand profiling products. In order to achieve this firstly a review of how demand is modelled within industrial practices outside of the Police will be presented, with consideration given to the suitability of attempting application of such modelling work to the Police problem. Description of the areas within the Policing system that are of primary interest will be described with particular emphasis given to definition of the demands upon staff members time and how this may be quantified; the understanding provided by this will be key in allowing construction of realistic demand models. This will lead to details of the construction of suitable demand models, including a Microsoft Excel based spreadsheet model making strong use of historical data and a C++ based model which constructs a profile using probabilistic methods. Methods by which staffing levels may be optimized in order to best meet a demand based objective function will be reviewed and then applied to demand profiles resulting from the models created. First among these methods will be tabu search, an optimizing technique which considers previous search iterations in order to provide diversification and encourage the investigation of a wide area of the potential solution space for a relatively low calculation cost. A second approach considered will be the implementation of a genetic algorithm based optimizing search methodology.

1.2 Demand Modelling

In many industries it is important to be able to gauge the amount of demand present for a particular product or service. Reasons for doing so may vary widely across different industries however the most commonly seen practices will be those involving estimation of profits

[2] or of expected service delivery levels [3]. Through clear understanding of the demands present within the Police industry it is possible to provide estimates of the staffing commitments required in order to satisfy them. This may then be assessed through an objective function in order to quantify either the savings to the Police organization or anticipated service delivery levels.

There are 2 primary methodologies by which demand my be modelled; these being the quantification of demand through some deterministic or semi-randomized process and alternatively through simulation of the process or system of interest. The first of these approaches seeks to make use of known data describing the system in order to immediately predict future system behaviour, through application of a suitable demand compilation function [4], [5]. The second approach focuses instead on creating a simulation that mimics the behaviour of the target system. Simulation may be more readily applied in situations wherein the system is understood but no function or deterministic method exists that may be used to quantify its behaviour. The most commonly seen simulation approaches make use of either discrete event simulation [6] or agent based simulation [7]. Discrete event simulation defines a series of chronologically occurring events which affect the state of a system and is usually defined to simulate the impact of events occurring over a period of time. In contrast agent based simulation instead defines a system and models the interaction of autonomous agents with this system or with each other.

1.3 Optimization Techniques

Optimization means to find the set of system elements from a potential space of all element combinations such that some objective function is either maximized or minimized. Methods that facilitate this optimizing process are known as optimization techniques. Of interest to this research are the linked targets of demand simulation and staff roster optimization, as such it is appropriate to consider the use of optimization techniques.

Within this thesis 3 search techniques are implemented in order to achieve staff roster optimization. The first of these is the tabu search algorithm, a metaheuristic local search method that makes use of logs of historically conducted search iterations in guiding the path of optimization [8], [9], [10]. The second method applied is that of genetic algorithms, an evolutionary algorithm search heuristic inspired by the theory of natural evolution [12], [13], [14]. Genetic algorithms effectively breed a population of solutions by some method in order to produce a new, stronger population. This is achieved through weighting of solution selection rates for breeding based upon their relative strength as assessed by objective function.

In order to prevent particularly strong local solutions dominating the search results, diversifying techniques based on natural processes such as mutation are implemented. Finally a hybridization of tabu search and genetic algorithms will be considered. Other researchers have noted the potential benefit to be gained from such an approach as each method is able to either compensate for or actively benefit from the weaknesses of the other [15], [16], [17].

1.4 Aims and Objectives of the Research

Through this research the primary aims are:

- To improve the ability within Leicestershire Police to match available resources to the demands for service, leading to improved Police response capability.
- To allow cost savings to be realized that may be reinvested into other areas within the Police organization.

To achieve these aims, there are several objectives to the research presented in this thesis, these are:

- To investigate the demands present upon the Police organization of Leicestershire, with particular attention paid to those demands relevant to front line officers.
- Assessment of a demand profiling tool created by Leicestershire Police in order to attempt quantification of demands on front line officers. Leading to the updating of this tool or the creation of an entirely new tool better suited to the intended task.
- Through application of the updated demand profiling tool or by other methods, to create realistic models of the demands upon Police staff.
- Application of operational research optimization techniques to allow the generation of staffing rosters that are better suited to meeting the defined expected demands.

Through use of operational research techniques it is desired to create a methodology capable of:

- Finding the most optimal distribution of available staff members across the shifts of a roster.
- Defining the shifts that are contained within a roster to better meet the target objective function.

- Allow variation in overall staffing levels such that increases or decreases in total staff employment levels may be considered.

1.5 Chapter Summary

This thesis is comprised of 8 chapters, these are:

- **Chapter 1. Introduction.** This chapter provides an introduction to the topics of this research and highlights the main aims and objectives. A summary of all chapters of this research is presented, describing briefly the key content and features of each.

- **Chapter 2. Literature Review.** The literature review describes research and application of demand profiling techniques in problems for industries other than the Police business. In particular, the nurse rostering problem, flow shop scheduling problem and stock management problems are discussed. Detail is provided of optimization techniques that were considered for application as part of this research but were not implemented due to the use of other more well suited techniques. These unused techniques are simulated annealing and ant colony optimization.

- **Chapter 3. Demand in the UK Police Industry.** Within this chapter the problem as posed by Leicestershire Police is described in detail, focusing upon a staff rostering problem for front line officers. A Microsoft Excel based tool created by Leicestershire Police is discussed and improvements to this are detailed, resulting in the creation of a new tool as part of this research.

- **Chapter 4. Tabu Search in Detail and Basic Implementation.** The tabu search optimization methodology is explained in detail. A basic application of this optimization technique upon the Police staff rostering problem is implemented and a series of tests are conducted upon the basic key factors defining the search in order to find best operating parameters.

- **Chapter 5. Tabu Search Advanced Implementation.** Diversification and intensification techniques are introduced and a few such techniques are created and tested in compilation with the best defined tabu search from Chapter 4.

- **Chapter 6. Genetic Algorithm Implementation.** Genetic algorithms are introduced with detailed explanation of the basic methodology and mechanisms that underpin their operation. A genetic algorithm contained within a piece of commercial software named modeFRONTIER is tested and shown to not be suited to the problem presented. Further to this, the creation and testing of a C++ based algorithm tailored to the Police staff rostering problem is detailed.

- **Chapter 7. Additional Work.** Several pieces of further work that have been conducted as part of this research are presented. These include consideration of a probabilistic demand profiling methodology in which statistical variation is included with the deterministic model, an objective function based on incident attendance rate targets and consideration of a hybridization of genetic algorithms with tabu search. Additional work concerning the incident profiling software developed for Leicestershire Police is discussed and a profiler describing demand upon canine trained units is presented.

- **Chapter 8. Conclusions and Future Work.** A summary of all work presented in this thesis is provided with the major conclusions drawn from each chapter. The objectives of the research are discussed with regard to the ability with which this research has met these objectives. Potential future work is detailed that could extend the scope of the research.

Chapter 2. Literature Review

2.1 Introduction

In order to assess the current state of demand modelling and optimization it is important to review work conducted by other researchers within this field. Thus this chapter presents the major information gained from such a review, detailing first applications of demand modelling within rostering problems, flowshop problems and stock management problems. Further to this, 5 optimization techniques are discussed and assessed regarding their relative suitability to implementation upon the staff rostering problem that will be defined for Leciestershire Police.

2.2 Demand Modelling in Industry

Many companies and industries are subjected to demand upon their products or services; in order to help with resource allocation a great many models have been developed ([18], [19], [20] for example) which cover a broad spectrum of industries of which the most relevant to this research are those that act in solution of staff rostering problems. This section presents 3 types of problem faced within industry and details work by other researchers towards their solution. These problems are staff rostering in the nursing and healthcare industry, flowshop scheduling in industrial manufacture and stock management in retail business. Further included is discussion of the similarities and differences between the types of demand experienced by these industries and that experienced by the Police force.

2.2.1 Nurse Rostering

The problem of shift scheduling for staff with varying levels of ability has been researched previously with the well documented nurse rostering problem [21], [22], [23]. In this problem it is very usual that nursing staff for a single ward are considered with their individual shift patterns and capabilities noted for potential optimization. Of key importance within the nursing industry when investigating potential automated rostering systems is the calculation time required in order to provide results of a suitable quality. This is due to the setting of hospital minimum staffing requirements so as to provide a health service to patients meeting or surpassing minimum expectations. These expectations will be defined by the availability levels of hospital facilities to demanding patients; however in addition more objective means such as assessment of waiting time for service may equally be considered.

This highlights the primary similarity between rostering staff for nursing duties and rostering policing staff. The Police organization makes regular use of 'minimum cover' figures for responsive policing staff that are maintained in order to attempt to meet the needs of the general public. However, the minimum cover figures themselves are an assumption and have no scientific backing or other support beyond being an estimation of required staffing levels. It will therefore be of importance to evaluate the currently used minimum cover figures in order to assess the matching between them and predicted or historically recorded levels of demand.

In the health care industry the working abilities of individual staff are important as it is possible for some members to be assigned multiple types of duty; for example a head nurse may work in place of a regular nurse for the duration of a shift (who may in turn work in place of a nurse aid) [24]. Within the Police force however many tasks require specifically trained staff and accordingly the behaviour of covering shifts between differently trained personnel is much more uncommon. This will provide an additional constraint upon the number of staff available for rostering within the work presented in this thesis that is not present for the nurse rostering problem.

Because the nursing staff may be switched it is very often the case that many changes to ward shift patterns are possible and it is necessary to define some form of objective function whereby the relative benefits of each pattern may be assessed. In order to achieve creation of this objective function and also to ensure that practical and effective rostering solutions may be generated a series of constraints are imposed; these constraints being of 2 distinct types, either 'hard' or 'soft'. Hard constraints are those that must be satisfied in order for a shift pattern to meet the bare minimum of service requirements; these include constraints such as minimum staffing levels and also restraints on how many hours an individual worker may complete in a given time period. Soft constraints are not critical and functional shift patterns will still be created if they are violated; examples of such constraints include individual worker preferences for particular shifts and targeted service levels above the minimum. Thus to optimize a shift pattern it is sought to firstly ensure that all hard constraints have been met and only after this to then attempt to meet as many of the soft constraints as possible [25]. In this way an objective function may be created by assigning scores to each of the soft constraints and keeping track of how many times each is violated in construction of a shift pattern; thus if comparing 2 patterns the one assigned the lowest score would be considered better.

Through usage of the tabu search method and genetic algorithms, much development and progress has been made into this field of optimization problems. The first major developments made use of computer search methods in order to replace the previous manual shift allocation procedure [24]. The research conducted focused upon the specific problem of staff shift pattern generation for Belgian hospitals and resulted in the creation of a commercial nurse rostering product, named Plane. Of interest for the purposes of this work are the methods used in order to create a successful product. Primarily a tabu search algorithm was used in order to explore variation in shift allocation to staff members, however results were further improved through the implementation of search heuristics. These heuristics were individually tailored to suit different needs and provide optimization taking into account a variety of perspectives. For example one heuristic was concerned solely with improvement of the worst shift pattern for an individual staff member whereas others sought to improve the visual representation of the shift pattern in as human a way as possible.

As an example of the effectiveness of the tabu search algorithm when applied to the nurse rostering problem and the potential improvements that may be made to staff rostering consider Table 2.1 illustrating the results of work conducted by Burke, Causmaecker and Berghe [24]. Results are shown from the application of 4 different search routines applied to 2 different problems created by consideration of real nurse staffing data; the methods used are a steepest descent algorithm, a tabu search method and 2 tabu search diversifications. The column category 'R_min' indicates optimization in which only the minimum set of system objectives are considered (hard constraints alone) whilst 'R_pref' refers to searches in which preferential (soft) constraints are also considered. The first diversification assumes that a member of staff can work either both or neither days of any given weekend; this was selected as when graphically represented within the final software it was found that incomplete weekends were the most noticeable feature for potential adjustment. The second diversification evaluates the roster for each individual staff member and seeks to improve the worst personal schedule through swapping shifts between it and another staff member's schedule. The objective function used in order to provide a 'value' for each roster is a compilation of penalty functions that look for undesirable features and score them according to severity. Thus a perfect roster that contains no undesirable features would have a value of 0, indeed the higher the score that a roster attains the less desirable it is.

It is concluded from these results that automated shift rostering systems provide great potential for improvements to staff and resource allocation. Further it is stated that for the case of the nurse rostering problem the tabu search method is a very well suited technique providing high quality staff rosters in a practical amount of time; these claims being assessed both by the objective function of the algorithm and through validation by resource plan-

Table 2.1: Results from application of tabu search to the nurse rostering problem

	R_min		R_pref	
	$Value$	$Time$	$Value$	$Time$
Problem1				
Steepest Descent	2594	$1'26$	2395	$1'37$
Tabu Search	2435	$2'05$	2214	$2'06$
Diversification 1	1341	$6'00$	1089	$5'59$
Diversification 2	1264	$20'15$	1011	$24'39$
Problem2				
Steepest Descent	1338	$0'44$	1338	$0'45$
Tabu Search	1189	$0'57$	1189	$0'58$
Diversification 1	843	$3'18$	843	$3'18$
Diversification 2	809	$6'25$	809	$6'25$

ning personnel. This strongly supports the notion of application of the tabu search method within the rostering of policing staff also due to the similarities present between the nature of the service that both organizations provide. There are however many differences in the underlying structures and the soft and hard constraints present in both systems, as such a great deal of investigation and customization of the search algorithm would be necessary in order to provide a useful solution.

In the work of Naudin et al. [26] a general staff rostering problem is considered and the researchers construct 3 models leading to its solution; these models each considering a different level of tasking that is assigned to each staff member. The first seeks to assign individual tasks on a moment by moment basis to staff, the second considers instead the assignation of daily rosters as a key variable and finally the third considers weekly rosters. A branch and bound methodology is applied to each of these approaches and conclusions are drawn about their relative strengths and weaknesses.

2.2.2 Flowshop Problems

In a great many industrial processes emphasis is commonly placed primarily on either the amount of time in which a unit of work may be completed or the amount of units that may be completed within a specified length of time. The common structure present in the vast majority of cases investigated is that of an industrial manufacturing process, wherein a series of tasks are required to be completed in a particular order through the usage of a variety of different machines. The notion of work flow capacity in this sense may at first appear to be of no considerable worth to the policing service, however there are potential benefits to be gained from such considerations in certain areas of the Police organization; for example the flow of prisoners held in custody suites [27].

The type of problem is the most commonly encountered and lends itself very well to solution by use of genetic algorithms and tabu search, see for example [12], [28]. In both of the referenced research documents a genetic algorithm is constructed and used in order to solve the n-job, m-machine permutation problem. This problem is designed to simulate the situation in which n jobs are required to be completed using m machines with the overall objective of finding a permutation of jobs that minimizes the total time required for single or batch product manufacture; this total time is commonly referred to as the makespan. Though not of direct use within the field of demand profiling for Leicestershire Constabulary there are other fields of the policing industry that could benefit from the application of such analysis; the effects of which could in turn be anticipated to provide some positive bonus to management of Local Police Officer (LPO) demand. For example both the Call Management Centre (CMC) and the custody suites follow a set of necessary procedures (jobs) and have a limited amount of staff (machines) in order to accomplish them; investigation into this is however beyond the scope of this research.

Murata, Ishibuchi and Tanaka [12] construct a genetic algorithm for the problem with the objective set as minimization of the makespan. This algorithm is compared firstly against several other known search procedures and then separately against hybridizations with other search algorithms. Through computer simulation it is shown that a genetic algorithm making use of two-point crossover and mutation based upon shift change is the most effective in solution of the defined problem. Table 2.2 details the results of the first series of tests comparing the resulting system makespans from the constructed genetic algorithm, local search, tabu search and simulated annealing.

These results highlight the potential strength of each of the methods used and indeed it is often seen within the literature that tabu search and genetic algorithms are very strong in application to a wide variety of problems.

2.2.3 Stock Management

It is common in business and, in particular retail, that effective stock management and prediction tools are sought. Though there are many inter-relations between flowshop problems and stock management the overall goals of stock modelling are not the same and approaches taken to achieve those goals differ considerably.

Table 2.2: Comparison of makespans from use of a genetic algorithm and other optimizing search methods

	Evaluations		
	10000	50000	200000
20Jobs			
Genetic Algorithm	101.5	101.0	100.7
Local Search	101.2	100.5	100.2
Tabu Search	101.1	100.5	100.0
Simulated Annealing	100.9	100.2	100.0
50Jobs			
Genetic Algorithm	102.3	101.4	101.1
Local Search	101.9	101.2	100.7
Tabu Search	101.4	101.0	100.5
Simulated Annealing	101.2	100.4	100.0

Stock management is concerned primarily with ensuring that at any given time the supply of a product is equal or greater than the demand for it. With such a clear impact on the profitability of individual products and entire companies it is of no surprise that a great amount of research and development has been conducted in this field [29], [30], [31]. The referenced paper of Silver et al. [29] provides a discussion of operations research and its applications within the field of inventory management. It further proposes a cataloguing system for stock management problems, going on to classify and describe some of the more well documented examples. An addition to the proposed catalogue scheme which would enable demand modelling problems similar to that found for Leicestershire Police to be included would be the inclusion of an additional type of self regenerating stock. This would mean that an individual stock item would be purchased by a customer and after a duration then become available for further purchase; this is understood more clearly in the example of policing to demand by considering that each officer will be able to respond to several unique demanding incidents within a shift.

There is similarity between the view of demand in inventory management and the way in which demand is perceived within the Police force; with the need for responsive officers taken as demand and the amount of officers available as supply. However it is usual for inventory management to work using a larger timescale, managing demand on a daily basis for example, whereas it is necessary for the Police to be adaptive and adjust the supply to meet a much more rapidly changing demand profile. The most pertinent similarities occur with the desire to predict future demand behaviour, based on historical records and anticipation of the effects borne out from fixed or predictable future events.

Predictive models use information obtained regarding historical system behaviour in order to effectively assess how the future of the system will act or develop. This is accomplished usually through several different analytical approaches that are used to view either the system in question as a whole or to scrutinize individual system components and measure their individual effects upon system performance. Within the Police force there is a wealth of historically recorded data as each officer activity, occurring incident and any other potentially useful information is recorded and logged; this is due to the importance of having accurate records of such information not only for Police records but additionally for use in a court of law. This robust database lends itself well to the creation and implementation of predictive modelling techniques.

2.3 Optimization techniques

Tabu search and genetic algorithms were used extensively throughout this research and detail regarding the specifics of their implementation is included alongside details of development and results in later chapters. Details on tabu search are included within Chapters 4 and 5 whereas information regarding the use of genetic algorithms in this research is included within Chapter 6. The implementation methodology behind other optimization techniques that were considered for use but not implemented within this research are discussed in more detail in this section; namely these techniques are local neighbourhood search, simulated annealing and ant colony optimization. Further, references to research within the literature using these other techniques as well as tabu search and genetic algorithms are discussed within this section.

2.3.1 Basic Neighbourhood Search

One of the simplest optimizing search techniques to implement, a basic neighbourhood search will investigate a selection of possible system states that are within some neighbourhood of the current state at each iteration of the search and select one of them based on the amount of improvement to objective function quality that it provides. The neighbourhood is often defined as the region of the solution space that may be reached through application of a particular transformation to the current system state. For example such a transformation could be the perturbation of the starting time of a single shift within the defined roster by a unit amount; in this example the neighbourhood would contain all rosters that have a single shift start time differing by a unit amount from the current roster. Termination of the neighbourhood search is commonly specified to occur at the point at which there are no improving system states available within the generated neighbourhood of the current solution.

This process may be summarized as shown in the flow diagram of Figure 2.1

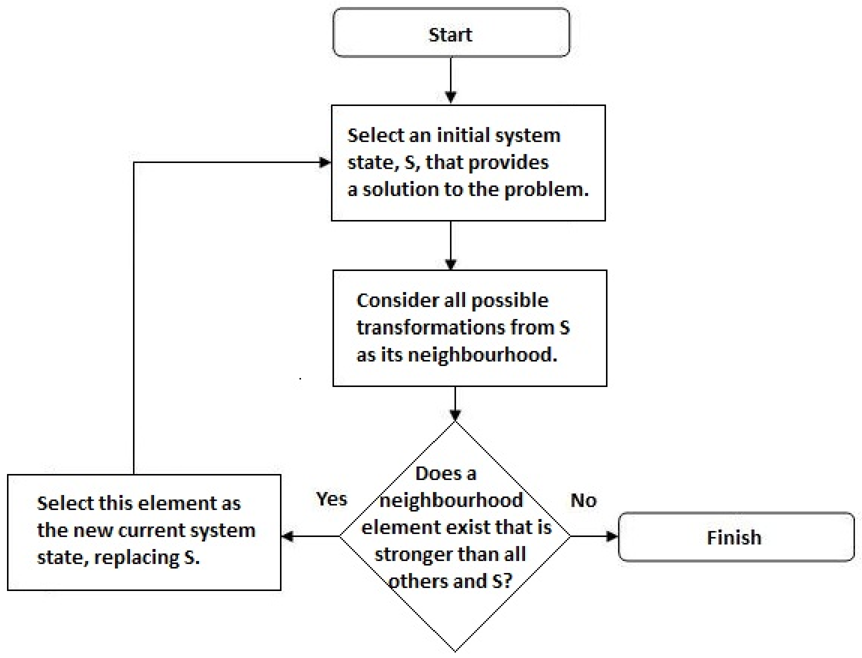

Start
Select an initial system state, S, that provides a solution to the problem.
Consider all possible transformations from S as its neighbourhood.
Does a neighbourhood element exist that is stronger than all others and S?
Yes
No
Select this element as the new current system state, replacing S.
Finish

Figure 2.1: Flow diagram describing basic neighbourhood search.

The method by which any particular candidate state within the neighbourhood is selected for use as the next state on the path to optimization may be varied to allow a less restricted search to be performed. A standard technique is to simply record the strength of every member of the neighbourhood through quantification by a suited objective function. Using these strength scores, a best neighbouring state may be identified as the one that provides the greatest improvement to the score attributed to the current system state and the search may then proceed by iteratively selecting the best found state in each newly defined neighbourhood.

An alternative candidate selection method may be employed that allows for semi-randomization of the state selected to become the next adopted in the process towards optimization. This is achieved through consideration of each candidate available for selection on an individual basis with the probability of selection being dependent upon the objective function value that is provides. Importantly the method should be designed such that a more improving solution is likely to be selected over a lessing improving result; in this way the search should still tend strongly towards an optimum. This approach allows some level of diversity to be introduced into the optimization process and should help discourage the discovery of a first available strongly local optimum. Indeed repetition of the search in a scenario with enough local optima should provide discovery of several differing results whilst the randomizing nature of the selection process is in effect.

2.3.2 Neighbourhood Search in the Literature

The basic neighbourhood search methodology is not often cited alone within the literature and indeed is commonly used as a comparative method by which the strength of other optimizing search approaches are assessed. The reason for this is that the theory behind neighbourhood search forms the basis of many other search algorithms, such as simulated annealing and tabu search [32]. Indeed it is seen that the base neighbourhood search technique is regularly outperformed by the other methodologies which are based upon it [32], [33]. Thus it is natural to select basic neighbourhood search in a similar comparative nature within this research in order to provide a benchmark for solution quality from which the effectiveness of other techniques applied may be assessed.

2.3.3 Simulated Annealing

Simulated annealing is a variation upon the local neighbourhood search approach with the major difference that a probabilistic technique is used instead of methodically searching for an improving solution at each search iteration. The simulated annealing process considers that each possible solution to a system is representative of some state of a physical system. Through perturbation of an initially poor system state (as quantified through some objective function) the search seeks to find an optimal state from the set of all those potential system states in the entire solution space. In this way it can be seen that the space upon which simulated annealing is most suited for use is that of a discrete solution space where there is some upper limit to the amount of possible system states that may occur (even if this amount is very high). Though the search is able to provide a global optimum, it could be expected that to do so would take longer than to exhaustively investigate each possible system state. Thus the most common objective with application of simulated annealing is in the discovery of a solution that meets some defined quality target within an acceptable amount of calculation time.

The search algorithm itself performs by, at each search iteration, considering some neighbouring state of the current system state and using a probabilistic approach in order to decided whether to move to this new state or to remain at the current state. Neighbouring states are those states that could be produced through some particular perturbation to the current state. For example, considering a defined series of shifts each with particular staffing allocations to be a system state then one method of generating neighbours could be through the movement of a staff member from one defined shift to another. Demonstration of this is included in Figure 2.2 wherein an initial system state (a) is shown and a potential neighbour (b) is generated by moving a staff member from the 1^{st} shift to the 3^{rd}.

Shift number	Staff Amount
1	5
2	3
3	0
4	2
5	3

(a)

Shift number	Staff Amount
1	4
2	3
3	1
4	2
5	3

(b)

Figure 2.2: A demonstration of system state perturbation in neighbourhood search.

The probability of selection for any generated neighbouring state to be accepted as the new system state is given by an acceptance probability function that is based on the objective scores of both the current and neighbour states and also on a global parameter called the temperature This function may be written as $P(x, x', T)$, where x and x' represent the objective values of the current and neighbour states respectively and T is a time dependent variable that represents the temperature. There are many requirements of this function that control the way in which it will act upon the search and encourage it to tend towards system optimization. Firstly it is important that the probability be non-zero when x' is greater than x; this allows the search to make non-improving moves, which helps in prevention of the search becoming trapped at a strongly attracting local optimum. Though this is the case it is also important that the probability by sufficiently small when x' is greater than x so that uphill moves are not so common as to cause the search to behave too randomly. This is accomplished through use of the temperature value which is initially set to some high value and then decreases as iterations of the search are conducted through some annealing schedule. As the temperature decreases, the probability of selection in cases in which $x' > x$ decreases and tends towards 0; so once T reaches 0 the search will only perform downhill moves. Finally, it was originally considered that the probability of selection should always be 1 in cases where x is greater than x' (that is, the neighbour has a better score than the current state) however this is not critical for the method to work and this condition is removed in some later work.

2.3.4 Simulated Annealing in the Literature

Simulated annealing has been used fairly extensively in application upon staff rostering and scheduling problems. Indeed the work of Murata at al. [12] mentioned in Section 2.2.2 makes consideration of a simulated annealing technique and compares solution quality of rosters generated from it against those provided by tabu search. The results of their work are illustrated in Figure 2.2 where it is seen that simulated annealing is one of the best performing amongst the optimization techniques tested. Further work of interest within this field conducted by Seckiner et al. [34] considers a job rotation scheduling problem in which some level of service is required to be delivered continuously over an extended time period. This is similar in nature to the Police organization wherein it is necessary to provide a service to the general public at all hours of the day on any day of the year; indeed this is the case for many public service industries in addition to some private ones. Further similarity to the Police organization is evident through the use of time varying demand upon services, with certain times of day and days of the week specified to experience peaks in demand. A series of trials are performed using their tailored simulated annealing approach and comparing it with the integer programming method of Carnahan et al. [35]; a summary of some key values from their results are presented here in Table 2.3. It is concluded from these results that though simulated annealing is slightly outperformed in most cases in terms of solution quality, it does greatly outperform the integer programming method in calculation cost.

Table 2.3: Results from comparison of integer programming and simulated annealing.

Problem	IP	SA	IP Time	SA Time
1	910	950	86400	238.36
2	730	760	86400	110.02
5	760	800	86400	40.39
8	314	311	86400	30.78

The result unfortunately indicates that though the calculation time is far shorter for the simulated annealing approach, the solution quality is lower than that provided by other methods (namely tabu search). Due to this it would be more suitable within this research to consider application of the tabu search method within this research instead of simulated annealing. There are no strong time constraints specified within the problem defined by Leicestershire Police and indeed staff rostering decisions are commonly made 10 weeks in advance of the time at which they are implemented.

2.3.5 Ant Colony Optimization

Ant colony optimization originated is based upon the known behaviours of ants as a probabilistic technique used in order to solve path finding problems in graphs and was first used by M. Dorigo in his thesis on learning and natural algorithms [36]. The method essentially simulates the movement of 'ants' upon the desired graph for optimization with the previous movement of each ant being recorded through a 'pheromone trail' that is left in its wake. All ants will originate from a single source and will attempt to reach a certain target position and then return to this source. In doing so, the pheromone trail left by an individual ant will dictate the path that was taken by it in both reaching the target and returning. Further ants will then leave the origin point with the same goal however their path will be less random and may instead be guided by previously left pheromone trails; this will result in the most well travelled pathways receiving the most pheromone trails and thus being more popular for selection as a pathway by future ants. The path is optimized through application of decay over time for each pheromone trail that is placed meaning that a path which takes longer in order to reach the target and return will have a weaker pheromone coverage than one taking less time to travel. Indeed with ants leaving the source at a constant rate, it would be seen that the pheromone trail associated with the shortest path would become strongest as more ants would follow it within a given time period. After some amount of time it would be expected that a particular pathway will be defined through the graph with a strong enough pheromone trail that all further ants will always travel upon this path; such a path should be the optimum and all other paths becoming untravelled would decay completely.

The graph travelled by the ants will be defined through a series of nodes that share links between them of varying length, as an ant reaches any particular node in the graph it will decide which path it should take based on any pheromone trails present and a desirability factor. This desirability factor is commonly set to be the reciprocal of the distance between the 2 nodes of the graph that a path links. The probability of a path between nodes i and j of a graph being selected for travel is given by

$$P_{i,j} = \frac{T_{i,j}^{a} D_{i,j}^{b}}{\Sigma(T_{i,j}^{a} D_{i,j}^{b}}$$

where T represents the pheromone strength on the path, D represents desirability, a is a factor that governs the importance placed upon pheromones and b governs importance placed upon desirability.

One of the particular strengths of ant colony optimization is that it may be run dynamically, upon a system wherein changes to the system structure are seen to be dependent upon time; this is a particular advantage over the usual applications seen with approaches such as simulated annealing, tabu search and genetic algorithms. Indeed this dynamic trait lends itself very strongly to the solution of routing and transportation problems for urban environments where it is common for the rate of flow to change dramatically at certain times; for example in simulation of transport to avoid traffic jams [37].

2.3.6 Ant Colony Optimization in the Literature

This methodology is strongly suited to application in those problems where some improvement to quality of solution may be obtained through optimization of a transport or routing scenario [37], [38]. Indeed Bell et al. [38] remark that not only is ant colony optimization capable of solving known vehicle routing problems to near optimal solution but it is able to do so within a comparable computation time as that required by other methods. Vehicle routing problems are traditionally defined such that there exist a set of resource hubs from which resources may be deployed in response to calls for service from customers; an example of this is provided in Figure 2.3 wherein a single hub is shown and required to supply resources to 5 customers. The algorithm applied seeks to find routes between customers such that all may be supplied with a resource whilst also optimizing some objective function; typical functions include minimization of total distance required to travel or minimization of total time required. There is some similarity here between this and deployment of Police personnel in attendance of incidents generated from calls by the general public. However with the time dependent nature of Police incident occurrence, it would rarely be the case that enough incidents exist in such a fashion for a single Police resource to be allocated a route to meet them all. This is due to the fact that many occurring incidents will be highly sensitive and will require a rapid response, coupled with large variation in the times that incidents will occur this leads to a level of unpredictability that is not suited for solution via ant colony optimization.

Ant colony optimization has also been applied successfully within non-transport related industries [39] [40]. Presented in the work of Gravel et al. is an ant colony optimization algorithm used to solve a scheduling problem for aluminium casting [39]. In the paper individual simulation ants are used in order to represent separate routes through the machine processing environment. Feasible solution constraints are enforced upon each route, such that only those that are able to provide the required series of operations (those ants that travel through all required nodes) in a suitable order and within a bounded time are allowed. Optimization then occurs through consideration of additional factors such as total resource

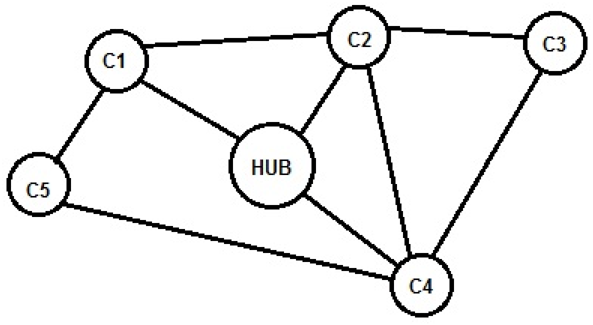


Figure 2.3: Example of a vehicle routing problem.

usage or total time usage within the process. The pheromone trail left by each ant is updated with respect to a single objective and by its design encourages the selection of future routes that make use of highly profitable combinations of route elements.

The ant colony optimization methodology is however not directly suited to application upon combinatorial problems such as the staff rostering problem posed within this thesis. It is particularly strong when considering routing problems or those in which a process is defined, however this will not be the case in this work. Accordingly this method will not be applied within this research.

2.3.7 Tabu Search in the Literature

The tabu search method has been applied successfully across a broad variety of problems [21], [43], [10] where it is seen to perform strongly in comparison against other optimization techniques in all cases. In particular it has been applied to great effect to the problem of staff roster optimization [9], [21], [11]. In the work of Dowsland [11] a tabu search optimization approach is applied to the specific problem of nurse scheduling. A dual target for optimization is provided through use of a method called strategic oscillation. By this technique the search first attempts to find a feasible solution that provides the cover required in order to meet all defined nursing requirements, upon discovery of such a result the search then changes objective (or oscillates) to instead consider the individual staff preferences with a roster. The end result from this is to generate a roster that is both feasible and also meets as many preferential objectives as possible. The strengths of rosters generated automatically are compared against those created by a human expert and both are seen to be of a similar quality as assessed by the staff preference based objective function. Importantly however, the human expert requires a great deal more time in order to produce such results than the automated tabu search method.

In the work conducted by Musliu et al. [44] a tabu search method is implemented that redefines the shift structure iteratively in order to best satisfy a given objective function. They show that such a process is of strong benefit to staff rostering problems and demonstrate its effectiveness on both generated and real world problems. Gaspero et al. [45] demonstrate an alternative objective to that considered here, instead seeking to generate staffing rosters with as few shift variations as possible. The intent of this is to reduce the complexity present within a series of defined shifts in order to increase the ability with which it may be managed. In contrast, the research presented within this thesis considers a fixed number of shifts and seeks to best fit available staff across them.

Tabu search has been seen to be very well suited to the combinatorial staff rostering problem, both in consideration of staff distribution across a defined roster and through iterative redefinition of the shifts composing a roster. These are precisely the factors that will be considered to form the basis of the optimization problem defined within this research and as such tabu search is a very suitable optimizing technique.

2.3.8 Genetic Algorithms in the Literature

The theory of the genetic algorithms search heuristic is widely applied across many fields of research, both industrial [14], [46], [12] and academic [47], [48]. The primary focus of academic research is in improvement to the algorithm when applied to general problems and through creation of new techniques and modifications that are able to increase the power of the search technique. Srinivas et al. [47] and Goldberg et al. [48] both make good example of this through consideration of refinement through the methodology by which solutions are selected at each iteration of the search. In particular Srinivas et al. investigate the effectiveness of allowing adaptive variation in the mutation and crossover selection rates used within the search that are adjusted based upon the strength of generated solutions, thus high quality solutions are more commonly preserved and lower quality solutions are more likely to be subjected to diversification.

Within this thesis, emphasis is placed primarily upon the application of optimizing search techniques to a defined series of staff rostering problems. Indeed as with tabu search, the method of genetic algorithms has been applied to the popular nurse rostering problem with a good degree of success [49]. Moz et al. [49] use a slight adaptation to the standard defined problem through consideration of a nurse re-rostering; a situation in which a staff member is suddenly unavailable to work a given shift, resulting in a required restructuring of the staffing roster. Their work was trialled using real world data from a Lisbon hospital and is noted as being able to provide high quality solutions in a suitably short amount of time. Indeed with the short notice at which these problems would arise in the real world it is of strong interest to be able to generate a compensating solution roster as quickly as possible. Ernst el al. [50] review a wide range of staff scheduling and rostering problems and note in particular the successful application of genetic algorithms that have been seen by other researchers in the field.

The noted ability of genetic algorithms to solve staff rostering (and indeed re-rostering) problems within a short amount of time suggests that they will be well suited in a similar role for the Police industry. Indeed with public safety and security as the primary goal of the Police organization, the ability to quickly generate suitable staffing rosters for deployment would be of particular benefit.

2.4 Summary

Though several optimization techniques have been discussed within this chapter, only 2 of them stand out as being most relevant for implementation upon the type of staff rostering problem that is defined by Leicestershire Police. These are the tabu search and genetic algorithm methodologies. With a wide array of applications on similar problems in the literature having been presented in both cases it is likely that they will also be well suited for this research. The basic neighbourhood search methodology however is noted as being suitable to provide a benchmark solution from which the scale of the improvements expected from application of tabu search and genetic algorithms may be assessed.

Chapter 3. Demand in the UK Police Force

3.1 Introduction

This chapter will provide an overview of demand factors relevant to front line policing staff, in particular Local Police Officers (or LPOs), and will demonstrate how these may be compiled to present a reliable image of the overall demand faced. A demand profiling tool originally created by Leicestershire Police will be discussed and flaws in its functionality and ability to realistically represent demand will be demonstrated. Further to this the development of a new profiler created as part of this research will be presented that overcomes the flaws of its predecessor. This enhanced profiler has been successfully used in order to inform new roster creation decisions within Leicestershire Police, highlighting potential risks with any given roster and allowing adjustment to better suit anticipated demand.

3.2 The Police Industry

3.2.1 Working Time Regulations

The working time directive [41] was implemented in the United Kingdom in October 1998 in an effort to regulate the statutory working time provisions for workers. The regulations laid out therein provide a framework that employers must adhere to when allocating working hours to personnel. A series of restrictions and limitations on the ability to manipulate staff shift patterns will thus be generated from this source; these being of critical importance with the goals of this research being such manipulations.

The most important considerations for the purposes of this research are those directly relating to the acceptable placement and timing of individual shifts. The most impacting amongst these being the requirement for staff to receive a specified minimum amount of rest and break periods over a duration of working time. In particular it is stated that no worker should be assigned more than 48 hours work within an average 7 day week; that is unless otherwise arranged through a suitable agreement between employer and employee. The average weekly value is usually calculated taking a staffing roster that covers a 13 week period in order to allow more complex situations to be suitably accounted for; such situations are encountered when a staffing roster is seen to change on a weekly basis. It is acceptable to use varying sample lengths where appropriate, for example the most commonly used staffing rosters within Leicestershire Constabulary are comprised of a 10 week recurring loop; in this case it is appropriate to calculate average weekly working hours over a 10 week period instead of a 13 week period.

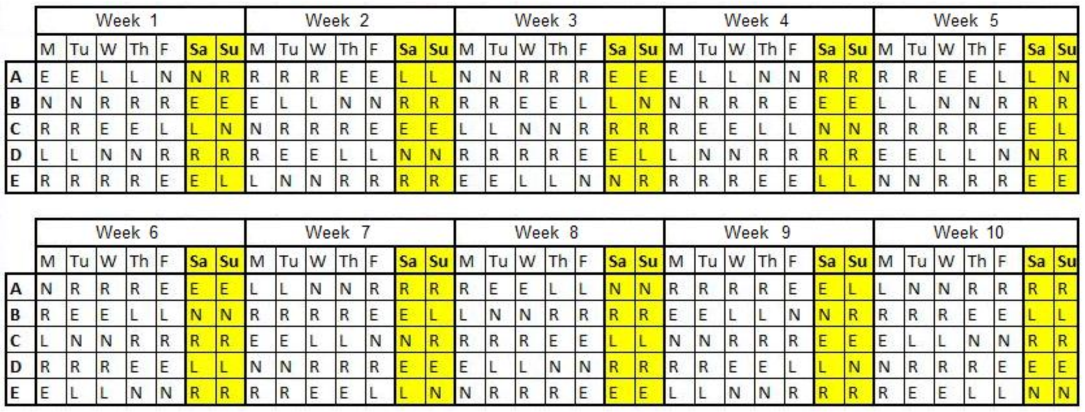
In certain cases how particular types of shift are defined is of interest when considering the variation in their structure; for example in making changes to the starting time and duration of individual shifts. Night shifts are most strictly defined in this way with the coupled requirements that they should be of at least 7 hours in length and must also cover the period between midnight and 5am. Under these constraints it may not be possible to make changes to the current night shift structure; however the situation is much more relaxed with respect to early and late shifts with much greater potential benefit to be expected from manipulation of these. An early shift may be of any length up to 12 hours and must regularly begin at some point in time between 7am and 10am. Late shifts essentially cover all remaining shift possibilities, but are most commonly seen to start between 1pm and 6pm. A lack of flexibility is further compacted by the definition of a working day in Leicestershire Police as starting and finishing at 7am; thus a night shift can at the latest finish at 7am and an early shift can at the earliest begin at 7am.

A standard full time worker should not then, on a regular basis, be allocated shifts which result in a schedule of more than 48 hours working time in a week. Working time being defined within the working time directive [41] as the hours in which an individual employee is contributing towards the productivity or benefit of their employer. Notably this definition does not include any time spent travelling to and from a regular place of work and accordingly an employee is expected to make their own arrangements in order to arrive on time for the start of a shift. Note that it is possible for a worker to opt out of the regulation and work more than 48 hours in a week, again through a suitable arrangement between themselves and their employer.

It is necessary that ample provision be made for break and rest periods within any given shift pattern; these are quantified through inclusion of 2 conditions. Firstly it is required that within any 24 hour period there be an undisturbed rest period of at least 11 hours. This period can however run between calendar days; indeed this is the commonest placement of such a rest period if standard office hour careers are considered. This rest period is to ensure that staff do not become overworked and have sufficient time for non work activities. The second condition is for any shift of length greater than 6 hours that a 30 minute break period be made available; though it is not stated whether a shift of 12 or more hours would necessitate 2 such breaks and for this research it is assumed that this is not the case. Both of these conditions are subject to the possibility of opting out; meaning that through a suitable agreement with their employer a worker may opt to have smaller break or rest periods. For health and safety reasons however this behaviour is not to be encouraged, particularly not for an extended period of time.

3.2.2 Shift Structure

The Leicestershire force has made use of the same, very basic, shift structure for a great many years and as such this is an area with great potential for adaptation in order to better allow demand to be satisfied. In full accordance with the working time directive [41] officer shift patterns are based around a 10 day structure; with 2 early shifts followed by 2 late shifts then 2 night shifts and finally 4 rest days. This pattern does not hold indefinitely however and there are regularly occurring spare shifts included within the staff roster that allow for additional staff cover on anticipated busy days. The Local Police Officers (LPOs) are divided into 5 shift groups labelled from A to E and these groups follow the same shift pattern with a time based staggering to ensure officer availability for each shift type on any day of the week. The basic pattern as employed by Leicestershire Constabulary is described in Figure 3.1; the figure illustrates the full 10 week cycle for each of the 5 shift groups all starting at the same point in time. The reasoning for using 5 groups to distribute the available staff is that it provides the simplest way of creating a recurring pattern from the 10 day structure described.



	Week 1							Week 2							Week 3							Week 4							Week 5						
	M	Tu	W	Th	F	Sa	Su	M	Tu	W	Th	F	Sa	Su	M	Tu	W	Th	F	Sa	Su	M	Tu	W	Th	F	Sa	Su	M	Tu	W	Th	F	Sa	Su
A	E	E	L	L	N	N	R	R	R	R	E	E	L	L	N	N	R	R	R	E	E	E	L	L	N	N	R	R	R	R	E	E	L	L	N
B	N	N	R	R	R	E	E	E	L	L	N	N	R	R	R	R	E	E	L	L	N	N	R	R	R	E	E	E	L	L	N	N	R	R	R
C	R	R	E	E	L	L	N	N	R	R	R	E	E	E	L	L	N	N	R	R	R	R	E	E	L	L	N	N	R	R	R	R	E	E	L
D	L	L	N	N	R	R	R	R	E	E	L	L	N	N	R	R	R	R	E	E	L	L	N	N	R	R	R	R	E	E	L	L	N	N	R
E	R	R	R	R	E	E	L	L	N	N	R	R	R	R	E	E	L	L	N	N	R	R	R	R	E	E	L	L	N	N	R	R	R	E	E

	Week 6							Week 7							Week 8							Week 9							Week 10						
	M	Tu	W	Th	F	Sa	Su	M	Tu	W	Th	F	Sa	Su	M	Tu	W	Th	F	Sa	Su	M	Tu	W	Th	F	Sa	Su	M	Tu	W	Th	F	Sa	Su
A	N	R	R	R	E	E	E	L	L	N	N	R	R	R	R	E	E	L	L	N	N	R	R	R	R	E	E	L	L	N	N	R	R	R	R
B	R	E	E	L	L	N	N	R	R	R	R	E	E	L	L	N	N	R	R	R	R	E	E	L	L	N	N	R	R	R	R	E	E	L	L
C	L	N	N	R	R	R	R	E	E	L	L	N	N	R	R	R	R	E	E	L	L	N	N	R	R	R	E	E	E	L	L	N	N	R	R
D	R	R	R	E	E	L	L	N	N	R	R	R	E	E	E	L	L	N	N	R	R	R	R	E	E	L	L	N	N	R	R	R	E	E	E
E	E	L	L	N	N	R	R	R	R	E	E	L	L	N	N	R	R	R	E	E	E	L	L	N	N	R	R	R	R	E	E	L	L	N	N

Figure 3.1: Current standard Leicestershire Constabulary shift pattern. E denotes an early shift, L a late shift, N a night shift and R a rest day.

This approach results in an average of 39.6 working hours per week when considered over the entire shift pattern, falling well below the maximum allowable figure of 48 hours defined in the working time directive [41]. In order to ensure that a minimum of 11 rest hours are given between shifts the only transitions are from an early shift to a late shift and from a late shift to a night shift; this usually provides 23 and 24 hours of rest respectively. It should be noted that the specifics of each type of shift can vary depending on the weekday upon which it falls; Table 3.1 details all such variations for the standard shift pattern.

Table 3.1: Variation in shift properties based on day of the week

Shift	**Days**	**Details**
$Early$	$Sun - Thu$	$7am - 4pm$
$Early$	$Fri - Sat$	$9am - 6pm$
$Late$	$Mon - Thu$	$3pm - Midnight$
$Late$	$Fri - Sat$	$5pm - 3am$
$Late$	Sun	$4pm - 11pm$
$Night$	$Mon - Sun$	$10pm - 7am$

Therefore it is of interest to consider this standard shift pattern as a benchmark by which any proposed adjustments and variations may be rated in their ability to allocate the staffing resources available in order to better meet demand.

3.2.3 Incident Grading Policy

There are a very large amount of calls for service from the general public that occur within the Leicestershire Policing region; 519043 such incidents occurred Countywide in the 87 week period from April 1^{st} 2007 to November 30^{th} 2008. These will fall into 4 grades based on the immediacy with which an officer will need to respond to the incident.

Firstly there are Grade 1 incidents, also known within Leicestershire Constabulary as emergency contacts. These are incidents that will result in an immediate Police response with the intention of officer arrival within 15 minutes of the Police being made aware; indeed a Countywide target has been set to arrive within this time for at least 85% of such incidents. Emergency contacts cover a wide range of circumstances and include situations in which an incident being reported contains, or is likely to contain, risk of the following factors:

- Danger to life,
- Use, or immediate threat of use, of violence,
- Serious injury to a person,
- Serious damage to property.

Further in cases relating to alleged criminal conduct an incident will be classified as an emergency contact for these reasons:

- The crime is, or is likely to be serious and in progress,

- An offender has been disturbed at the scene,
- An offender has been detained and poses, or is likely to pose, a risk to other people.

Finally traffic incidents requiring the attendance of policing staff may also be classified as emergency contacts if either of the following criteria are filled:

- It involves or is likely to involve serious personal injury,
- The road is blocked or there is a dangerous or excessive build up of traffic.

Secondly in the grading system are Grade 2, priority response, incidents. Again these have a targeted officer response period, this time set at 60 minutes, and a County objective of 80% attendance of such incidents within this time. Grade 2 incidents are considered those for which any of the following statements are applicable:

- There is genuine concern for somebody's' safety,
- An offender has been detained but poses no risk to others,
- A witness or other evidence is likely to be lost,
- At a road collision, there are injuries or a serious obstruction,
- A person involved is suffering extreme distress or is otherwise deemed to be extremely vulnerable,
- Force policy dictates a priority response (e.g. distraction burglary),
- Hate crime.

Next are Grade 3, or scheduled response, incidents. These are those in which a rapid or immediate Police presence is not necessary and it would be more suitable to arrange a time with the individual reporting the incident at which suitable action could be undertaken. The only reason given on the Leicestershire Police incident grading policy for which this may be the case is that the response time is not critical in apprehending offenders. These incidents may be resolved either through an officer visiting the reporting individual at a convenient time or the individual themselves visiting their local Police station in order to further the report. For example this could be the result of a reported vandalism where the witness would like to provide a statement however they are currently unavailable to do so; thus appointing a later time for this to happen.

The final category is that for Grade 4 incidents which are accomplished through resolution without deployment. The grading policy states that resolution without deployment is appropriate where the needs of the caller can be adequately met through telephone advice or Service Delivery Desk intervention, through access to the Force and National Frequently Asked Questions databases, the involvement of another more appropriate agency or service or through some other method.

For the purposes of this research, primary focus has been upon the grade 1 and 2 incident demand as these are the incidents that require swift or immediate response by major front-line Police officers. The impact of grade 3 incidents is considered, however incidents of this grade may be scheduled for response at times to suit both the involved individual and a Police staff member.

3.2.4 Demand on LPOs (Local Police Officers)

3.2.4.1 Definition of Demand The demand on Local Police Officers (LPOs) is compiled from a number of factors, however the primary part of the service they deliver (and indeed the one that their performance is most strongly assessed upon) is that of response to calls for service from the general public. Records of these calls are maintained by the Police Call Management Centre (CMC) and contain a wealth of information describing many key features of every incident. This is certainly an important task as an accurate and reliable recorded history may be crucial for reference at a later date both for internal Police purposes and for use in a court of law. As an example of the numbers of such calls for service received, Table 3.2 contains demonstrative example hourly figures for the total amount of grade 1 to 3 calls received that result in incidents within a specified region of Leicestershire over the period from April 1st 2007 to November 30th 2008. These are categorized by hour of the day so, for example, between the times of 07:00 and 07:59 on a Sunday there were recorded a total of 57 incidents; each column is headed by a value that indicates the hour of each day in which incidents occur.

These 3 grades of incident are all those that will require some type of response from Police staff. Thus considering this data it is possible to build up at the least a simplistic view of demand that may outline a general pattern within the demand experienced by LPO staff. Clearly however the time at which a call is received will not always be precisely the same as that at which a Police unit is assigned in order to resolve it; this is especially true of grade 3 incidents where an appointment is scheduled to allow for resolution at a convenient time. This highlights the 2 overall types of incident based demand experienced by LPOs, responsive demand that is typically composed of grade 1 and 2 incidents and scheduled demand

Table 3.2: Total amount of Grade 1-3 incidents occurring in the CB region over the period from 01-04-07 to 30-11-08.

Day	00	01	02	03	04	05	06	07	08	09	10	11
Sun	242	213	188	121	90	75	46	57	77	119	136	186
Mon	179	89	78	36	33	33	43	68	117	182	177	203
Tue	155	112	75	62	29	34	41	63	125	156	188	181
Wed	123	108	77	52	29	34	30	68	121	167	178	217
Thu	148	93	66	60	32	34	32	62	118	178	167	205
Fri	166	95	77	58	38	30	43	75	95	162	173	214
Sat	285	214	149	139	98	71	52	52	107	124	173	196

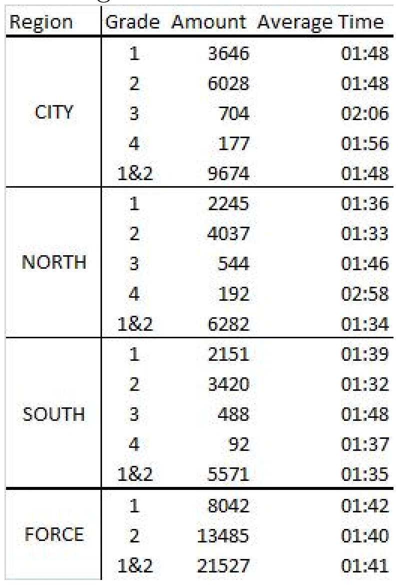
Day	12	13	14	15	16	17	18	19	20	21	22	23
Sun	196	239	270	309	270	313	312	355	303	290	231	207
Mon	189	220	268	282	301	361	375	375	342	286	236	215
Tue	209	243	231	274	309	358	357	353	360	259	279	227
Wed	193	219	254	272	266	328	350	386	339	281	282	185
Thu	222	214	228	257	315	309	318	352	344	282	242	194
Fri	172	203	210	268	299	325	359	382	381	354	379	367
Sat	217	227	220	280	293	284	288	308	348	328	285	323

mainly comprised of grade 3 incident resolution.

It is possible to look further into the particular types of incident occurring within each grade by considering the categorizations employed by Leicestershire Police in incident definition. A series of 5 major categories are defined, with incidents each being allocated to one of them at the initial point of contact between a Police call management centre staff member and the reporting member of the public. These 5 categories are anti-social behaviour, crime, public safety, road related and other.

3.2.4.2 Influence from Recent Demand Considering only the amount of incidents historically occurring within a given period of time will not be capable of rendering a truly accurate representation of demand. Indeed every incident that occurs will require the attendance of at least 1 Police officer for an amount of time that will vary widely based on its nature and severity. An emergency Road Traffic Collision (RTC) for example will require at least 2 officers; 1 of these will ensure safety by closing roads, redirecting traffic, etc. and the other will deal with the scene itself. Depending on the severity of this type of incident even more staff could be required and the amount of time they are needed could be very high, taking several hours to fully resolve. In contrast a scheduled appointment in order to record a victim statement for a burglary offence could only need the attendance of a single officer for just 1 hour. To aid in illustration of this Table 3.3 describes the average times in hours and minutes that it takes for response to and resolution of incidents within each of the 3 Basic Command Units (North, South and City BCUs) and over the County as a whole. Also mentioned are the total number of incidents that were considered in generation of the average time figures; this data being drawn from a time period of 3 full calendar months.

Table 3.3: Average incident 'assigned to released' times based on BCU and grade.



Region	Grade	Amount	Average Time
CITY	1	3646	01:48
	2	6028	01:48
	3	704	02:06
	4	177	01:56
	1&2	9674	01:48
NORTH	1	2245	01:36
	2	4037	01:33
	3	544	01:46
	4	192	02:58
	1&2	6282	01:34
SOUTH	1	2151	01:39
	2	3420	01:32
	3	488	01:48
	4	92	01:37
	1&2	5571	01:35
FORCE	1	8042	01:42
	2	13485	01:40
	1&2	21527	01:41

Leicestershire Police maintain a record of all incidents attended by LPO staff on the computerized system called OIS; within this system logs are kept of, amongst much other data, the starting time and ending time of each incident. Using this data it is possible to define an 'assigned to released' time, this being the amount of time that passes between assignation of an officer to an incident and their future release from it upon resolution. Note it is also possible for an officer to be released before complete incident resolution; this may happen for a few reasons, for example the occurrence of a higher priority emergency incident or reclassification of the incident to a lower priority. The average 'assigned to released' time does not vary a great deal across the Leicestershire BCUs or even across the individual LPUs, which is contrary to what may have been anticipated due to the logistic differences between LPUs; in particular differences that would be thought to occur between urban and rural areas. This is shown in the average incident 'assigned to released' times within Table 3.3.

3.2.4.3 Double Crewing

It is possible that more than 1 officer may work together as a single policing unit for the duration of their shift; within Leicestershire Police this practice is known as double crewing. This may be represented by a reduction in the resources available for use in order to meet demand or equally it can instead be simulated through appropriate scaling of demand figures themselves. For example consider a situation in which there are 3 incidents requiring Police attendance at a single moment in time and that there are 5 officers available, working with 2 paired and 1 single; this is then equivalent to having 3 units available for these 3 incidents, 2 of the units being double crewed and 1 being single crewed. Similarly considering again that 3 incidents require Police attendance and that the ratio of double to single crewed units is known, for example 2:1 (representing 2 double crewed units for each single crewed), it is possible to estimate the required number of staff officers that would be required to meet demand using this crewing policy ratio. This is achieved by constructing first a suitable factor and then multiplying it by the number of incidents needing attendance. The suitable factor for this 2:1 example would be 1.667 and is calculated as follows:

$$factor = 1 + \frac{double}{(double + single)} \tag{3.1}$$

Essentially then the fraction gives a value representing the chance that an incident will be attended by a double crewed unit. So in the 2:1 example, the chance of incident attendance by a double crewed unit is equal to $\frac{2}{(2+1)}$ or 0.667. Adding 1 to the factor is then suitable to create a multiplying factor that will scale demand values to indicate the number of staff members required using a given double crewing structure.

3.2.4.4 Directed Action Policing

This form of policing includes other demands on LPO time that are caused not from incidents received from calls by the general public but instead from active policing, such as National Intelligence Model (NIM) tasking and reassurance patrols. Due to the variability in the assigned levels of these duties between different LPUs it would be expected that demand due to such policing would be subject to a great amount of variations from region to region. Indeed there would be anticipated not only different approaches to directed action policing undertaken by each LPU but also varying expectations of it. Further such demand will vary in order to take into account special events such as national holidays and major public or sporting events held locally. Fortunately such demand is entirely predictable and indeed is generated as a result of planning and preparation by Police staff officers, hence this is not considered necessary for quantification within this research.

3.2.4.5 Custodial Incidents Of the incidents attended by Police officers many will result in the arrest of an individual or group of offenders. When this happens the arresting officers will be engaged with the processing of resulting arrestees through the custody procedure. The first stage of this process involves the arresting officer in transporting their charge to the nearest available custody suite. In some cases this will be part of the local Police station site; however in others they must instead travel a fair distance across the County in order to reach the nearest suitable site. Upon arrival at the custody suite the officer and offender will enter a queue, waiting to be processed by custody staff; the length of time spent waiting will vary depending on how busy the suite is and how many others are waiting in the same queue. After leaving this queue the arrestee will be signed in and recorded as becoming a guest of the suite; part of this signing in process involves the completion of paperwork which, with a compliant arrestee, will take approximately 20 minutes. Depending on the nature of the incident resulting in arrest a drug or alcohol test (or even both) may be required. Primarily this is to ensure an accurate record which can later be quoted in a court of law where necessary; a list of offences for which this will be necessary is included in Figure 3.2. If not previously recorded it will then be required to both fingerprint and take a DNA sample of the offender; this process will in the best case take a further 30 minutes. After completion of each of these steps, the arrestee will finally be signed over to the custody suite releasing the attending Police officer back to active duty.

Theft	Fraud
Begging	Attempted theft
Attempted fraud	Persistent begging
Robbery	Possession of article for use in fraud
Production of Class A	Attempted robbery
Supplying articles for use in fraud	Supply of Class A
Burglary	Handling stolen goods
Possession of Class A	Attempted Burglary
Attempted handling stolen goods	Possession of Class A to supply
Aggravated Burglary	Taking without owner's consent
Arrest on Inspector's authority	Aggravated vehicle taking
Equipped to commit offence	

Figure 3.2: List of offences for which drug and alcohol tests are required (sourced from DT1 record form).

Conjectural estimates of the total amount of time required to deal with a custodial incident vary widely across Leicestershire Constabulary; brief questioning with 8 officers (4 working in custody suites and 4 LPOs) provided a wide range from 3 hours to 9 hours with an average estimate of 5 hours. Indeed it is a fairly common assumption that an officer will be occupied for the remainder of their shift if they are required to deal with the transport and handling of an arrestee. From this it is possible that an arrestee will still be undergoing processing at the point in time that the shift of the supervising officer ends. When this occurs it is necessary for a judgement call to be made on the part of the shift Sergeant in command in order to decide either to authorize overtime for the currently attending officer or instead to transfer supervision of the prisoner to a different officer. This act of arrestee transference between officers at the end of a shift is known within the force as a handover. Most commonly handovers will be the transference of an arrestee from an officer that is at the end of their designated shift to another officer that is just starting their shift.

As an example of the scale of arrests from incidents that will be expected to occur within any given day, Figure 3.2 contains values describing the total amount of incidents recorded on the Police computer system OIS over the 87 week period from April 1st 2007 to November 30th 2008; with Table 3.4 separately containing those that resulted in an arrest over the same period. The data for both of these figures are taken from the same policing region allowing them to be compared; for example then, of the 57 incidents occurring between 07:00 and 07:59 on a Sunday, 4 were seen to result in an arrest. These figures may be used to calculate, for each hour of the day and day of the week, the percentage of incidents that have historically resulted in an arrest giving a guideline for anticipated future arrest rates. Doing this provides an average of 3.901% of incidents resulting in an arrest with a standard deviation of 2.281%, essentially meaning that it would be expected for future arrest rates to vary somewhere between approximately 2% and 6%. The maximum experienced rate of arrests within this period is 14.706%; though individually this is a statistically significant large value (when considering a 95% confidence interval hypothesis test) there are not enough other such significant values occurring with any pattern that could suggest an underlying structure to their appearance.

There is however the suggestion of a time period in which the percentage of arrests from incidents is commonly far below the overall mean value, this being the time period between 16:00 and 21:00. This time period is illustrated within the table shown in Table 3.5 which contains the percentage of incidents resulting in arrest by hour of the day and day of the week in which they occur. Indeed even within this indicated range there is still a single point at which arrest rates are significantly greater than the average, though with this only occurring for 1 out of 42 points of data it could be considered to be an anomalous point. This potentially commonly occurring period of decreased arrest incidence warrants further

Table 3.4: Total amount of live incident arrests occurring in the CB region over the period from 01-04-07 to 30-11-08.

Day	00	01	02	03	04	05	06	07	08	09	10	11
Sun	13	10	12	5	7	10	4	4	4	4	4	6
Mon	13	3	4	1	1	1	0	0	2	4	3	10
Tue	11	4	1	3	2	5	2	2	7	5	9	8
Wed	5	8	3	2	2	0	1	3	6	3	4	13
Thu	5	6	7	2	1	1	0	1	7	6	4	5
Fri	12	8	4	5	3	2	4	4	3	3	4	4
Sat	10	14	7	11	6	1	3	1	2	4	4	4

Day	12	13	14	15	16	17	18	19	20	21	22	23
Sun	11	3	6	7	0	9	6	4	5	10	17	11
Mon	9	10	8	6	7	11	6	11	7	7	10	14
Tue	6	5	3	9	11	17	15	10	4	6	9	10
Wed	10	4	9	10	6	10	11	3	7	9	11	7
Thu	7	16	15	9	9	12	8	11	7	11	13	10
Fri	8	5	6	3	11	9	4	14	8	11	9	11
Sat	9	8	9	9	9	10	6	7	6	12	14	20

investigation and accordingly its appearance in data compiled from the other policing regions should be investigated.

Table 3.6 describes the total significant variations seen in custodial incident occurrence per LPU both as a number of significant data points seen ($\#Sig.$) and as a percentage ($\%Sig.$). Unfortunately, as illustrated by the table, this region of significant low arrest rates is only present within the CB policing region and indeed the next fewest number of large valued points within the same time period is 4; this equating to 9.52% of the dataset. It is however plausible that there is some distinct difference in the CB region as opposed to the other LPUs which may cause this behaviour to be apparent only in that area. In considering creation of a model or other application to suit a general case, it should be considered most suitable to ignore the presence of such a low custodial incident rate region; instead taking the more commonly represented situation wherein the rate of incidents resulting in arrest fluctuates a small amount around a fixed value.

3.2.4.6 Administrative Demand A study on the work of LPOs conducted by Richard Ashton, a former technical management manager at the International Association of Chiefs of Police (IACP) Division of State and Provincial Police, has suggested that the average LPO

Table 3.5: Total amount of live incident arrests occurring in the CB region over the period from 01-04-07 to 30-11-08.

Day	00	01	02	03	04	05	06	07
Sun	5.37%	4.69%	6.38%	4.13%	7.78%	13.33%	8.70%	7.02%
Mon	7.26%	3.37%	5.13%	2.78%	3.03%	3.03%	0.00%	0.00%
Tue	7.10%	3.57%	1.33%	4.84%	6.90%	14.71%	4.88%	3.17%
Wed	4.07%	7.41%	3.90%	3.85%	6.90%	0.00%	3.33%	4.41%
Thu	3.38%	6.45%	10.61%	3.33%	3.13%	2.94%	0.00%	1.61%
Fri	7.23%	8.42%	5.19%	8.62%	7.89%	6.67%	9.30%	5.33%
Sat	3.51%	6.54%	4.70%	7.91%	6.12%	1.41%	5.77%	1.92%

Day	08	09	10	11	12	13	14	15
Sun	5.19%	3.36%	2.94%	3.23%	6.12%	5.44%	5.19%	4.85%
Mon	1.71%	2.20%	1.69%	3.23%	5.82%	1.36%	2.24%	2.48%
Tue	5.60%	3.21%	4.79%	4.42%	4.31%	4.12%	3.46%	2.19%
Wed	4.96%	1.80%	2.25%	5.99%	3.11%	2.28%	1.18%	3.31%
Thu	5.93%	3.37%	2.40%	2.44%	4.50%	1.87%	3.95%	3.89%
Fri	3.16%	1.85%	2.31%	1.87%	4.07%	7.88%	7.14%	3.36%
Sat	1.87%	3.23%	2.31%	2.04%	3.69%	2.20%	2.73%	1.07%

Day	16	17	18	19	20	21	22	23
Sun	5.93%	5.43%	5.77%	5.35%	6.60%	7.24%	9.52%	11.11%
Mon	0.00%	2.49%	1.60%	1.07%	1.46%	3.50%	7.20%	5.12%
Tue	2.27%	3.07%	1.68%	3.12%	1.94%	2.70%	3.58%	6.17%
Wed	4.14%	5.18%	4.29%	2.59%	1.18%	2.14%	3.19%	5.41%
Thu	1.90%	3.24%	3.46%	0.85%	2.03%	3.19%	4.55%	3.61%
Fri	3.01%	3.69%	2.23%	2.88%	1.84%	3.11%	3.43%	2.72%
Sat	3.75%	3.17%	1.39%	4.55%	2.30%	3.35%	3.16%	3.41%

Table 3.6: Significant custodial incident occurrences for each LPU.

LPU	# Sig.	% Sig.
CB	1	2.38%
CH	5	11.90%
CK	6	14.29%
CM	14	33.33%
CN	5	11.90%
CW	4	9.52%

LPU	# Sig.	% Sig.
NH	9	21.43%
NL	6	14.29%
NM	11	26.19%
NR	9	21.46%
NW	4	9.52%

LPU	# Sig.	% Sig.
SB	4	9.52%
SH	6	14.29%
SM	6	14.29%
SW	12	28.57%

will spend up to 40% of their shift partaking in administrative duties. This is however noted to include meal and toilet breaks in addition to what may be considered active working time such as attending briefings and carrying out paperwork. In addition to this it is concluded that 27% of an LPOs time will be spent travelling; though it is worth noting that this time will be composed for the most part of travel both to and from incident sites. Important to note is that incidents attended by LPOs recorded within the Police data system OIS will include not only the travelling time to an incident but indeed should also contain the majority of administrative paperwork associated with the same incident. In practice there are concerns that officers will sometimes be released from an incident and thus viewed as an available resource for redeployment whilst there is still incomplete paperwork remaining; policy dictates that an officer is only to be released upon completion of all associated paperwork.

3.2.4.7 Abstractions The Police business is no different from any other profession in that it carries with it the additional factor of staff abstraction. Such abstractions can be due to a large variety of things and may also be seen to last for practically any length of time. Due to this it is common practice for a business to identify a series of key abstraction categories which may be used in order to aid in the regulation of such events. Leicestershire Police make use of 2 major categories which are defined around the anticipated duration of an abstraction. These are Tier 1 abstractions which may last anything up to 28 days and Tier 2 abstractions which are defined as being of at least 28 days in duration.

One of the most frequently occurring causes for abstraction is sickness. This is a very broad category and may fall under either the heading of Tier 1 or Tier 2 and will depend on the particular circumstances; for example a case of flu may only last a few days or weeks whereas a serious injury followed by rehabilitation could last for several months. Indeed, cases of major injury may result in the involved officer becoming unable to return to active front line policing due to elevated risk of further injury or an inability to safely perform their prior duties. It is common practice in situations such as these to reassign the injured officer to a desk based role that makes use of their particular knowledge and skills.

Less common but still an important abstraction to consider is that from maternity and paternity leave, these affect nearly every industry and the Police business is no exception. The West Midlands Police Federation provide a guideline to aid in appropriate time being provided for maternity leave and state that up to 15 months should be available to be taken by an individual; this is further defined by allowing no more than 6 months to be taken before and 12 months taken after the due date of the infant. This length of time is a considerable amount for even a single officer to be made unavailable. Additional complexity is added through the fact that the individual may elect to return to their position upon the end of leave, making the long term employment of a replacement to fill the role in their absence much harder.

In addition to these, there is also abstraction from work due to annual leave which may total up to 28 days for each member of staff within the Police force. Such leave is however only granted through suitable application on an individual basis and may indeed be refused in some cases if resulting staffing levels would be deemed as too low. This is particularly common around the Christmas and other major holiday periods as many staff members will seek to obtain leave for this highly desirable time. However the Police business must always be functioning and it is impossible to allow all staff to take national holidays as leave; it has actually been claimed conjecturally by many members of Police staff that national holidays may commonly see elevated levels of demand and as such would require more staff than usual. Special events such as sporting fixtures and local events also commonly fall on days at which it is desirable for applications for annual leave; indeed the event itself may be the cause of high desirability. These days would also be anticipated to require elevated staffing levels and could thus also result in the refusal of annual leave requests. Indeed, because of certain events Leicestershire Police have been seen to state that no annual leave will be accepted for particular dates well in advance of their occurrence.

To help staff planners in estimating the amount of staff that should be expected to be abstracted at any one time an international standard of 70% attendance is assumed. This would mean that, for example, if it were desired that 7 staff members be on duty for any particular shift then planning personnel should assign 10 members under the assumption that up to 3 of them may be abstracted. Of course true abstraction levels will vary but the estimated 70% attendance rate is intended to provide a worst case scenario and it should be very rarely seen that true attendance levels fall below this.

3.2.5 Summary

The key factors affecting demand that have been identified within this section are:

- The rate at which incidents occur.
- The amount of time required in order to fully resolve an incident.
- Type and grade (or severity) of each occurring incident.
- A double crewing provision.
- Arrests resulting from live incidents.
- Any additional administrative time required to resolve an incident.
- Staff abstractions from rostered duties.

How each of these factors have been considered within both the original and newly developed incident profiling softwares will be discussed in the following sections.

3.3 Police Incident Profiling Software

Within this section an overview of the original Police demand profiling software will be given, detailing all key factors that are used within it to compile an image of the demand experienced by a Local Police Officer (LPO). This original was created as an in-house piece of software within Leicestershire Police force; all further work, research and development was conducted separately as part of this research. Weaknesses of this profiler are discussed; with approaches taken towards resolution of these issues displayed through the description of an improved demand profiling software. This improved software is currently made use of within Leicestershire Constabulary to aid in the estimation of demand levels upon LPOs.

3.3.1 The Original Demand Profiler

The original piece of demand profiling software created by Christopher Newbold, a member of Leicestershire Police staff, was designed as a Microsoft Excel spreadsheet collecting various relevant forms of data to provide an overall image of the demand upon LPOs. This software is well suited for use within the Police business as the Microsoft suite is widely available to staff members within the organization; not only this but the Excel package demonstrates much capability for use in this manner. Indeed it is a long process to follow in order to allow a new piece of software to be made available for use on machines connected to the Police internal network; it is effective to instead make best use of the software already freely available.

3.3.1.1 Base Demand Data The basic data upon which the demand profiler is built is generated from records of calls for service received by the Call Management Centre (CMC) from the general public. These records will store, for every incident, a large amount of important data; for example, the type of incident occurring, the collar numbers of attending officers, specific incident resolution notes, etc. Of interest for the purposes of the demand profiler however is the fact that these records will always contain the grade of an occurring incident, the starting time of the incident and the final resolution time of the incident. With these a basic image of demand from calls to service may be constructed as discussed in the following sections; this may then be further enhanced by consideration of other factors (such as double crewing and custodial incidents) to create an estimate of staffing levels required in order to meet total demand.

3.3.1.2 Time Segregation In order to construct a demand profile illustrating variation in incident demand using the data it was necessary to segregate incidents based upon the time of their occurrence. Initial work for the first Police demand profiler was conducted using a segregation period length of one hour; as an example with this period length, any incident occurring between 2:00am and 2:59am would be allocated to the 2am category. Figure 3.3 illustrates the result of this segregation for average levels of incidents requiring swift attendance (those incidents of grade 1 or 2) over one Local Policing Unit (LPU) in Leicestershire. In this illustration average hourly amounts of incidents requiring swift response are shown for a Monday in the chosen region of Leicestershire. The call based demand represented in this figure is shown to reach a minimum at around 6am in the day, slowly rising as the day progresses and peaking in the region of 7pm to 10pm; this is precisely accordant with the conjectural view of demand as understood by the average Police officer.

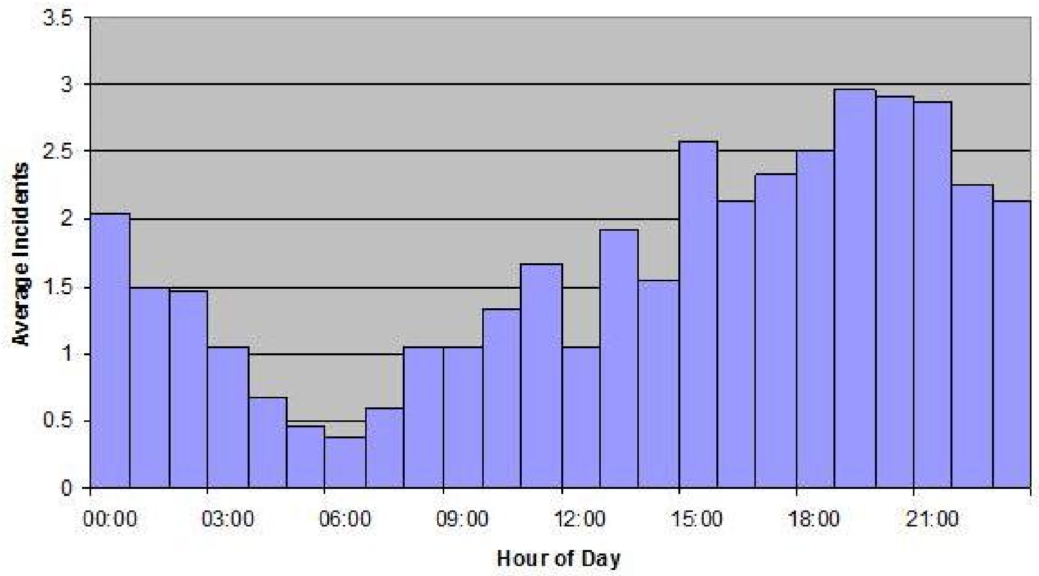


Figure 3.3: A histogram describing the average grade 1&2 incident levels occurring on a Monday in a sample policing region.

To increase the accuracy of the model and thus also improve the ability to demonstrate realistic demand profiles, it is necessary to further develop this concept through reduction of the time segregation period. However it must also be considered that segregation period reductions beyond a certain point would yield only marginally better results but require a much greater amount of time and effort in order to implement. Further, for particularly small period lengths it would be expected that the resulting periodic demand figures could become negligible and thus the potential to critically underestimate incident based demand could be introduced. This is especially true for typically low crime areas where over-reduction could feasibly predict that no incidents will occur at all. Changes to this are one of the key factors that occur within later stages of development of Police demand profiling software, resulting in a higher resolution image of demand. Within this first profiler however the time segregation period is fixed at 1 hour in length.

This segregation of the incident data recorded from calls to service by the general public is used to provide the base foundation from which the view of demand is constructed in the Police demand profiling tool. Other factors considered will cause adjustment to this either through manipulation of the base demand itself or through addition of independent demand from other sources. The factors of ‘influence from recent demand’ and ‘double crewing’ are both cases of base demand level manipulation; whereas ‘directed action policing’, ‘custody factor’ and ‘administrative factor’ are all examples of supplementary independent demand. How each of these demand influencing attributes are taken into account within the first Police demand profiler will be described and discussed individually.

3.3.1.3 Influence from Recent Demand With the historically recorded incident data categorized both by day of the week and by hour of the day the resulting representation shows only the volume of incidents that start in each hour. To truly represent the demand that is borne from these incidents it is important to also take into account the duration of each incident.

Primary focus was initially placed upon incidents requiring immediate or swift response (those being incidents of grades 1 and 2 respectively) and as such only the incidents in those categories were considered to contribute towards demand on an LPO. Using the data shown in Figure 3.3 it was decided that the average resolution time of 1 hour and 41 minutes for all incidents of grades 1 and 2 across the entire County would provide a useful guideline figure. However as the first demand profiler could only use incident lengths of some integer number of hours it was necessary to round this up to 2 full hours. This means that for each incident needing swift or immediate attendance that occurs, 2 full hours will be assigned within the profiler for it to be considered resolved. The potential for overestimation of demand due to this is quite high; increasing the resolution times for each incident from the average of 1 hour 41 minutes up to a full 2 hours represents an increase in the volume of demand by 18.8%.

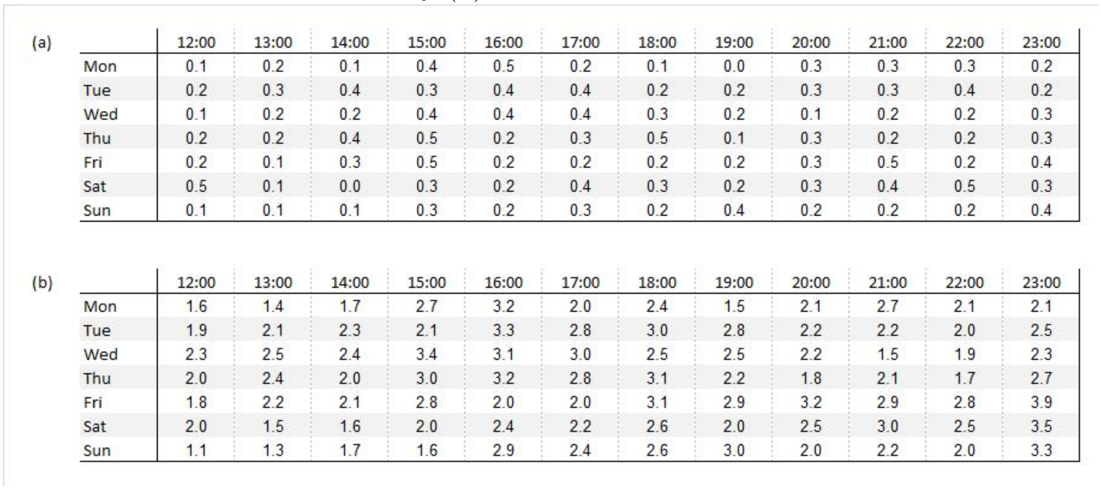
3.3.1.4 Double Crewing The demand profiling tool incorporates a double crewing feature capable of simulating the effects that such staff allocation will have. In the initial model a simplified mechanism has been implemented whereby double crewing is considered to be either on or off, meaning that either all officers are double crewed or all are single crewed. This is accomplished in the model through use of a double crewing factor by which all other demand is multiplied. In the basic form used in the first Police demand profiler the double crewing factor is set to a value of either 1 or 2 in order to represent single or double crewing respectively. This value is adjustable based on hour of the day and day of the week, thus select time periods may be identified as double crewed with all others being single crewed.

3.3.1.5 Custodial Incidents Within the original demand profiler version the custodial factor is applied in a similar way to that for the basic incident demand; hourly figures for occurring arrests are added directly to demand and are given relevance for a duration of 3 hours. This means that any incident resulting in an arrest will effectively add 3 hours to the resolution time for that incident within the demand profiler. The figure of 3 hours was derived primarily from conjectural evidence and consultation with informed members of Police staff. It is thus highly possible that improvements to accuracy and realism could result from development through a more detailed investigation of the custody process with respect to LPO attendance requirement.

The data used to indicate arrest occurrence levels for this factor is taken from records of arrests from live incidents; this meaning incident reports that are finalized using any of the closing tags used to indicate an arrest. Due to the nature of such incidents, they often occur infrequently and it is fairly common that within any chosen hour the historical average will be less than 1 incident. Because of this the original demand profiler made use of a boundary value at which rounding up occurs in order to ensure that custodial effects are not ignored. Within the original demand profiler this boundary value was set to be 0.09 incidents; this figure was chosen in order to demonstrate model functionality and with no scientific reasoning or other justification. Clearly then this is of great concern when considering validity of the model and the accuracy of custody factor representation within it; such a high degree of rounding would be expected to greatly overestimate the effects of custodial incidents in some cases. Table 3.7 displays the average recorded hourly incident levels as used within the original demand profiler both over an individual LPU (a) and the County as a whole (b).

It is seen from the information displayed in Table 3.7 that at the single LPU level (a) it is a very common event that average arrest levels are lower than 1 per hour. With the

Table 3.7: Average custody levels experienced hourly between noon and midnight for a single LPU (a) and the entire Leicestershire County (b).



(a)

	12:00	13:00	14:00	15:00	16:00	17:00	18:00	19:00	20:00	21:00	22:00	23:00
Mon	0.1	0.2	0.1	0.4	0.5	0.2	0.1	0.0	0.3	0.3	0.3	0.2
Tue	0.2	0.3	0.4	0.3	0.4	0.4	0.2	0.2	0.3	0.3	0.4	0.2
Wed	0.1	0.2	0.2	0.4	0.4	0.4	0.3	0.2	0.1	0.2	0.2	0.3
Thu	0.2	0.2	0.4	0.5	0.2	0.3	0.5	0.1	0.3	0.2	0.2	0.3
Fri	0.2	0.1	0.3	0.5	0.2	0.2	0.2	0.2	0.3	0.5	0.2	0.4
Sat	0.5	0.1	0.0	0.3	0.2	0.4	0.3	0.2	0.3	0.4	0.5	0.3
Sun	0.1	0.1	0.1	0.3	0.2	0.3	0.2	0.4	0.2	0.2	0.2	0.4

(b)

	12:00	13:00	14:00	15:00	16:00	17:00	18:00	19:00	20:00	21:00	22:00	23:00
Mon	1.6	1.4	1.7	2.7	3.2	2.0	2.4	1.5	2.1	2.7	2.1	2.1
Tue	1.9	2.1	2.3	2.1	3.3	2.8	3.0	2.8	2.2	2.2	2.0	2.5
Wed	2.3	2.5	2.4	3.4	3.1	3.0	2.5	2.5	2.2	1.5	1.9	2.3
Thu	2.0	2.4	2.0	3.0	3.2	2.8	3.1	2.2	1.8	2.1	1.7	2.7
Fri	1.8	2.2	2.1	2.8	2.0	2.0	3.1	2.9	3.2	2.9	2.8	3.9
Sat	2.0	1.5	1.6	2.0	2.4	2.2	2.6	2.0	2.5	3.0	2.5	3.5
Sun	1.1	1.3	1.7	1.6	2.9	2.4	2.6	3.0	2.0	2.2	2.0	3.3

demand profiler rounding any of these values which are higher than 0.09 up to 1 there can be seen great potential for overestimation; indeed for the LPU data shown this rounding would increase the total average arrests seen from 22.2 up to 82, an increase of 269.4%. This is a very large increase of the demand due to arrests experienced when the profiler is used to model a single LPU; the Countywide figures however are subject to no such rounding issues as all values are greater than 0.09.

3.3.1.6 Administrative Factor Considering the work study conducted by Richard Ashton and the resulting figure of 40% time spent daily on administrative duties, a factor was created for the Police demand profiler to show officer unavailability due to such duties. It was considered that some of this 40% would be constructively spent as part of the workload of an LPO. Indeed as the base demand type data is sourced from the record of incidents generated from calls to service by the general public it should be expected that some administrative work is already contained within this data. To clarify, as described in Section 3.3.1.3, the basic data takes account of the length of time between assignment and future release of Police staff members from an incident. This period of time will include that required to resolve all paperwork related to the incident itself; thus taking the whole 40% value would result in this segment of administrative duties being added to a resulting profile twice. With this in mind the administrative factor was set to a value of 25% for the original version of the demand profiling tool. This means that at any point in time it would be expected that 1 in 4 officers would be engaged with some type of administrative duty and as such would be unable to respond to a newly occurring incident.

Further investigation of the accuracy of this assumption has been considered and could involve repeating the work of Richard Ashton, with particular attention paid towards how this administrative factor breaks down with respect to ‘working administrative’ and ‘non-working administrative’ time. An alternative option would be to conduct a survey of willing LPOs in order to obtain perceived time spent on administrative duties. Though the survey would be expected to provide results of lower accuracy than a work study it does have the benefit of yielding a large amount of useful data requiring less effort and in a much shorter period of time.

Indeed many changes to this figure could have been expected as other initiatives already underway come into action. For example, the work on mobile data currently in progress will allow many administrative duties to be performed by officers on site; this will reduce not only administrative time but will also alleviate the need for an officer to return to the station between incidents and thus reduce time spent travelling. However it was found that administrative time should already be taken into account within the type of demand data used and as such demand enhancement due to administrative duty was removed from the profiler. The base demand data upon which a profile is based considers the length of each incident to be the amount of time from which it is assigned to an officer to the point at which the same officer is released and free for other duties. This time should include the conduction of all administrative duties (paperwork, travelling, etc.) associated with an incident.

3.3.1.7 Abstractions Internationally a standard attendance rate figure of 70% is assumed within most businesses or organizations when considering staffing allocations. This means that out of every 10 staff members that are assigned to work any given shift, only 7 are anticipated to actually attend work.

The original demand profiler makes use of this figure by considering a series of user input staffing levels to cover the time period of interest and then reducing these by 30% to simulate losses due to abstraction. By considering both the planned number of staff members and this reduced level it would be anticipated that the true staff attendance level should lie somewhere between the two. Figure 3.4 illustrates how this data is recorded within the demand profiler itself.

The user enters a series of values into the upper 3 white numeric boxes to indicate how many staff members they are allocating to each of the 3 shift categories; these early, late and night categories are defined bands of time within which a shift may start to be classified as one of the 3 types. The profiler then simply reduces this amount by 30% to provide the user with

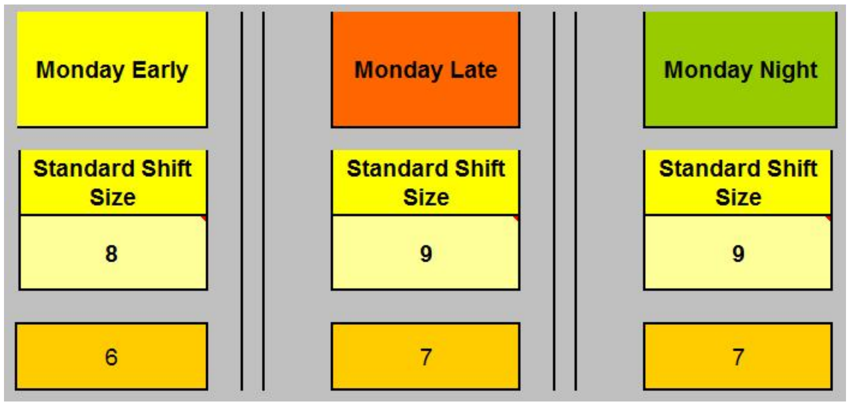


Figure 3.4: Shift size user input boxes and calculated staff after abstraction within the original profiler.

the number of staff they should anticipate as available in each of the shift categories in the grey numeric boxes. Within the original profiler there are 7 sets of these boxes to cater for each of the days of the week individually.

3.3.1.8 Shift Selection The levels of staff available within each of the categories early, late and night are then distributed by the user as they see fit across a variety of potential shift starting times. Times available within the original demand profiler are fixed and are based upon those suggested with the proposition of staggering shift starting times to better allow LPO staff to meet demands made upon them. Traditionally the standard LPO shift pattern will be formed by following the pattern of 2 early shifts, followed by 2 late shifts, then 2 night shifts and finally 4 rest days. In general each of the 3 shift types has a fixed start time, the demand profiler has a selection of times around these available for staff reallocation; the standard starts are 7am for an early shift, 3pm for a late shift and 10pm for a night shift. The original demand profiler makes use of the following 9 possible shift start times for each day of the week:

- Early Shifts: 7am, 9am, 11am
- Late Shifts: 1pm, 3pm, 5pm
- Night Shifts: 6pm, 8pm, 10pm

3.3.1.9 Compiling Demand All of the aforementioned information is collected together and used to calculate for every hour of a calendar week, what the average expected level of demand should be. This begins by first considering the total number of incidents of either grade 1 or 2 that will occur within each hour of the week; this is simple to calculate using the available historical records of incidents maintained by the Call Management Centre (CMC). Data is obtainable for the total number of incidents occurring for each hour of each day of the week historically, however this total will be for all days of the same type. For example, the total for Monday between 1:00am and 1:59am will be the total number of incidents between those times for all Mondays within the period covered by the dataset. To obtain an average from this then it is a simple matter of dividing the total figure given by the number of times that the appropriate day features in the dataset. Table 3.8 provides an example of this, displaying the total and average numbers of incidents occurring in a series of 1 hour periods on Sundays in a target region across a 153 week period. For example, over the entire 153 week period considered a total of 28 incidents were seen to occur on Sundays between 07:00 and 7:59; division of this by the number of weeks (153) provides an average incident occurrence rate of 0.183 for this time period on a Sunday.

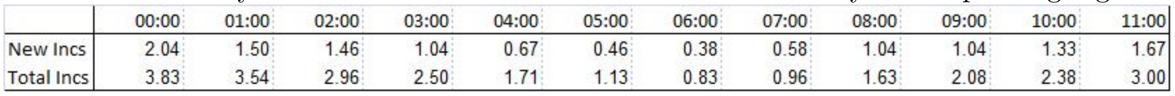
Table 3.8: Total and average occurring incidents in an example region over a 153 week period.

	Grade	07:00	08:00	09:00	10:00	11:00	12:00
TOTAL	1	28	26	20	28	27	23
	2	70	65	81	80	76	75
	3	22	20	27	39	28	41
AVERAGE	1	0.183	0.170	0.131	0.183	0.176	0.150
	2	0.458	0.425	0.529	0.523	0.497	0.490
	3	0.144	0.131	0.176	0.255	0.183	0.268

With the average number of incidents per hour requiring attendance of LPO staff now calculated it is important to consider the influence from other recent demand. This is borne from the amount of time that will be required in order to resolve each incident individually. As described within Section 3.3.1.3 it is assumed in this profiler that every incident will require 2 hours in order to resolve. Thus every incident that occurs within one hour will be marked as ongoing into the following hour where it will still be contributing to demand. This is illustrated in Table 3.9 wherein it is seen that the total incident demand for any hour is the sum of the amount of incidents that initialize within the hour itself and those initializing in the preceding hour; for example the total demand at 7am (0.96) is the sum of the newly occurring incidents at 6am and 7am (0.38 and 0.58 respectively).

The double crewing factor is now considered, this is whether or not LPO staff will be double crewed within vehicles or not; the effect of them being so will essentially double the number of officers required in order to meet demand. Thus for every hour the total incident demand

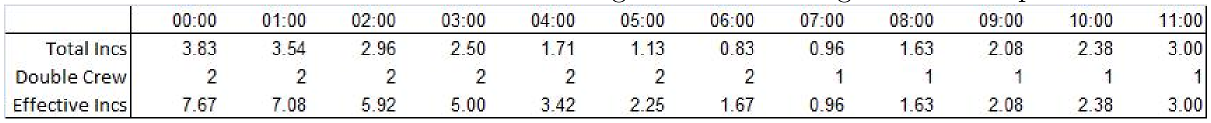
Table 3.9: Hourly base and total incident demand levels for a day in one policing region.



	00:00	01:00	02:00	03:00	04:00	05:00	06:00	07:00	08:00	09:00	10:00	11:00
New Incs	2.04	1.50	1.46	1.04	0.67	0.46	0.38	0.58	1.04	1.04	1.33	1.67
Total Incs	3.83	3.54	2.96	2.50	1.71	1.13	0.83	0.96	1.63	2.08	2.38	3.00

is doubled if staff for that hour are designated as double crewed. How this is recorded within the demand profiler is illustrated in Table 3.10. The double crewing factor is recorded as either the integer value 1 or 2 (to represent single and double crewing respectively) for each hour and used as a multiplier of the total incident number in order to simulate the effect of double crewing. Thus in Table 3.10 7:00am has a factor of 1 which indicates single crewing and so the effective incidents are the same as the total incidents; 4:00am is marked as double crewed with a factor of 2 and accordingly the effective incidents are seen to be twice that of the total incidents from the previous step shown in Figure 3.9.

Table 3.10: Effects of the double crewing factor in the original demand profiler.



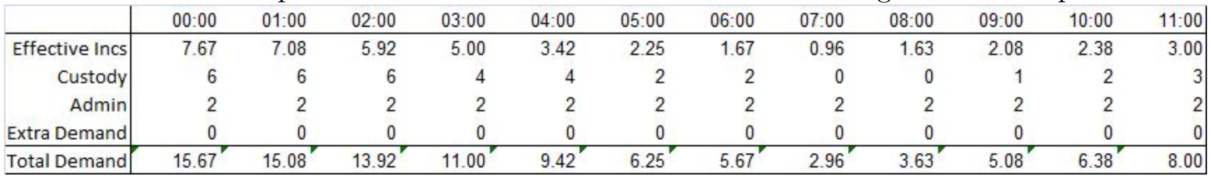
	00:00	01:00	02:00	03:00	04:00	05:00	06:00	07:00	08:00	09:00	10:00	11:00
Total Incs	3.83	3.54	2.96	2.50	1.71	1.13	0.83	0.96	1.63	2.08	2.38	3.00
Double Crew	2	2	2	2	2	2	2	1	1	1	1	1
Effective Incs	7.67	7.08	5.92	5.00	3.42	2.25	1.67	0.96	1.63	2.08	2.38	3.00

Custodial incidents are the next form of demand to be considered in the profiler and similarly to the basic demand data these are first calculated by collecting historical data to generate average rates of occurrence per hour. To consider the length of time that will be required in order to resolve an arrest fully (taken in this profiler to be 3 hours) these averages are then adjusted in a similar way as the base incident data when the influence from other recent incidents is included. The difference is that the effective arrest carry over length is 3 hours and not 2 as with the basic incidents. Thus the total number of custodial incidents occurring within any hour will be the sum of those that begin in the current hour and those that began in the preceding 2 hours. This total number of arrests per hour is then also multiplied by the same double crewing factor as previously used; officers when paired will remain so for the entirety of their shift in this model, so when a pair makes an arrest they are both needed to follow through the entire arrest process.

Administrative duties are assessed by considering the factor described in Section 3.3.1.6 whereby it is assumed that 25% of an LPOs time will be spent on administrative duties that remove them from the availability to respond to other calls for service. The first demand profiler utilizes this by essentially stating that 25% of an allocated staffing level will be unavailable at any given time. This amount of staff that are essentially otherwise engaged is recorded by an increase in demand levels and as such this results in demand being increased at each hour of the day by 25% of the staff assigned to the same hours.

These last 3 forms of demand bearing data are all then directly added on top of the current demand totals for each hour individually; this is represented within Table 3.11. The custody, administration and other additional demand figures are all added to the effective incident values from the previous stage of demand compilation. For example, at 4:00am there is demand for 4 officers to resolve custodial incidents, 2 officers to carry out administrative duties and 0 officers for other planned tasks; this raises total demand levels from 3.42 to 9.42.

Table 3.11: Compilation of additional demand factors in the original demand profiler.



	00:00	01:00	02:00	03:00	04:00	05:00	06:00	07:00	08:00	09:00	10:00	11:00
Effective Incs	7.67	7.08	5.92	5.00	3.42	2.25	1.67	0.96	1.63	2.08	2.38	3.00
Custody	6	6	6	4	4	2	2	0	0	1	2	3
Admin	2	2	2	2	2	2	2	2	2	2	2	2
Extra Demand	0	0	0	0	0	0	0	0	0	0	0	0
Total Demand	15.67	15.08	13.92	11.00	9.42	6.25	5.67	2.96	3.63	5.08	6.38	8.00

3.3.1.10 Graphical Representation To provide the user with a visualization of the supply and demand of resources a primary graphical output of the form illustrated in Figure 3.5 is given. Included upon this graph is the demand data for the associated day of the week alongside several different staffing levels. These staffing levels represent 4 different staff allocations; assigned staff on the standard shift pattern, assigned staff minus abstractions on the standard pattern, a user input distribution of staff amongst staggered shifts and an automatic distribution across the same shifts.

The lightly shaded area in Figure 3.5 represents the total calculated demand level for each hour of the day. Time on the graph starting at the upper vertical axis marker labelled 7; this represents 7am with time then progressing clockwise and increasing through the rest of Monday and up to 6:59am on Tuesday. Thus the graphical representation for Monday will cover the continuous time period from 7am Monday to 6:59am Tuesday. The outermost dark line illustrates the total assigned numbers of staff members when allocated to the standard LPO shift pattern; the dashed line represents the same but however takes into account the industry standard 30% abstraction rate. The inner continuous dark line indicates the custom staffing level input by the user, which in this example is showing slight reduction of staffing

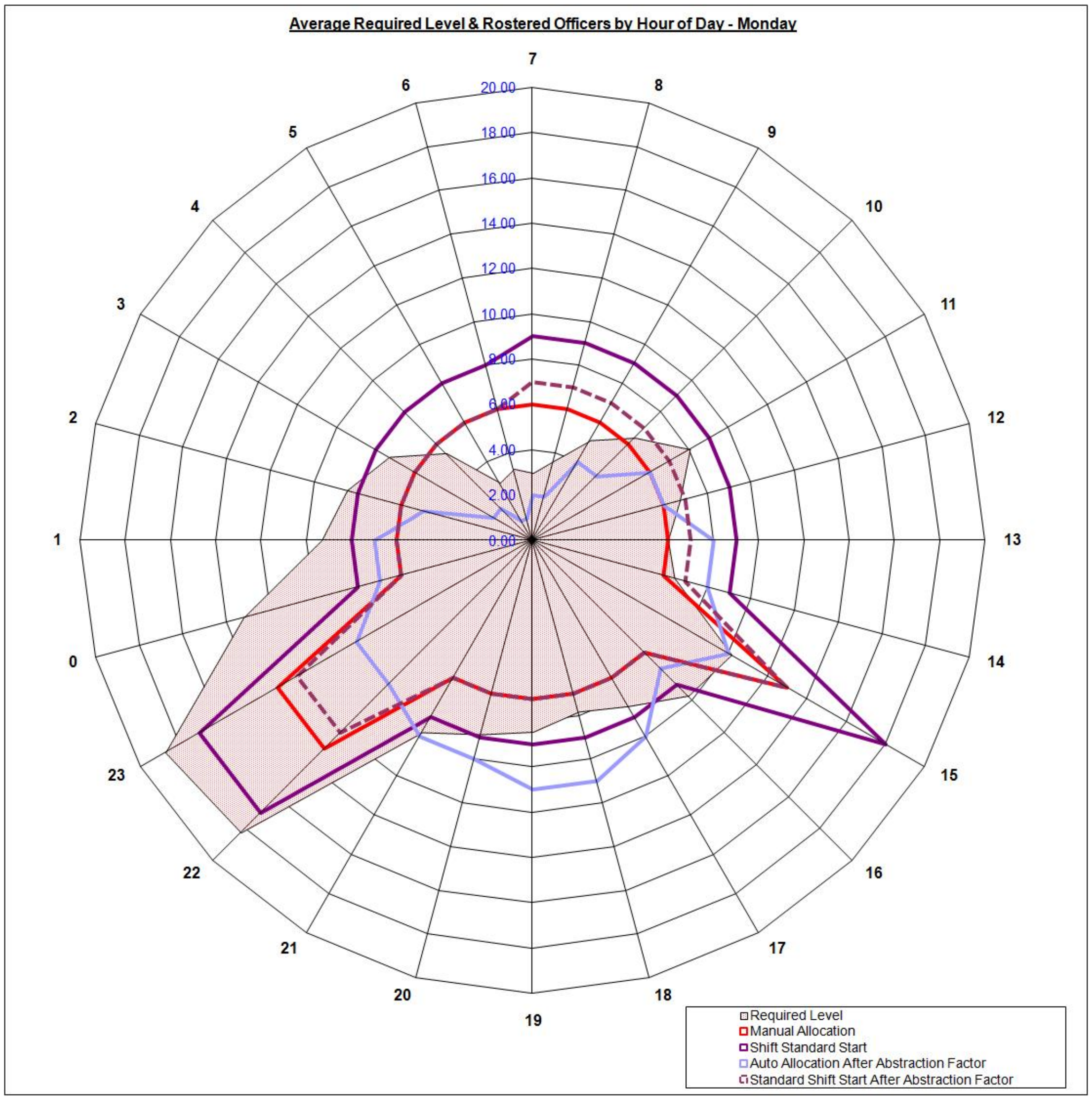


Figure 3.5: Sample graphical output of the original demand profiler software when used to simulate demand on one policing region.

below maximum anticipated abstraction. Lastly the faint inner line shows the automatic redistribution of staff performed by the profiler; this is accomplished by distributing available staff as evenly as possible across all 9 of the available shifts. Figure 3.5 can be seen to illustrate both an area in which overstaffing occurs with the standard shift pattern (between 3:00 and 15:00) and an area in which understaffing is seen (between 21:00 and 2:00).

Upon completion of full evaluation of the original demand profiling software it was sought to create a new profiler also within Microsoft Excel; this new version should not only tackle any weaknesses identified in the original but also make use of any enhancements thought to improve software capability.

3.3.2 Summary

Through this section a number of features catered for and factors used by the original profiler have been identified, these are:

- Incident resolution time for all types and grades of incidents set to 2 hours.
- Double crewing must be applied to either all or none of staff.
- A series of pre-specified starting times and shift durations are provided.
- Staff abstractions are applied to input total staffing amounts.
- Time spent on administrative work is considered.
- Directed action policing and planned Police activities may be included.

Further, through initial investigation and validation work conducted as part of this research upon the original incident profiler, a series of flaws were identified that could be resolved in order to improve accuracy and performance. The flaws identified to be addressed in Section 3.4 are:

- High increase in predicted demand levels from rounding caused by large data segregation periods.
- Limited range of shifts available for staff redistribution attempts.
- High level of rounding present for occurrence rates of custodial incidents.
- Incident administration time is dependent upon staffing level and not incident amounts.
- Double crewing is set such that all units must be double crewed or all must be single crewed for each individual shift.
- Abstraction rates are not strictly necessary in planning amount of staff required to meet demand.
- The base data set used for historical incident and arrest occurrence rates only covers a 5 month period.
- Staff entry is purely accomplished through user trial and error.

In terms of tackling these flaws, three stages of development have been investigated:

- Stage 1. Validation and adjustment of demand modeller inputs and factors.
- Stage 2. Investigation of statistically expected elevated demand levels.
- Stage 3. Application of simulation and optimization.

3.4 Profiler Development Stage 1. Validation and Adjustment of Inputs and Factors

3.4.1 Time Segregation

One of the greatest weaknesses within the original profiler was the high level of data rounding occurring due to the segregation of data into hourly categories; this was shown to lead to overestimation of demand both for the base incidents and for custodial incidents. In order to compensate for this it was sought to reduce the length of each segregation period to an appropriate value that would be capable of increasing the accuracy of final demand values presented by the model.

To begin work towards this goal the first step was in recalculating the average incident resolution times; this was necessary as the data set upon which these original results were based was limited to only that from 5 calendar months. The newer results were instead calculated using 12 months of recorded incident data such that average resolution times from a full calendar year would now be made. The results of this work were to discover that over an entire year the average resolution time for incidents of grades 1 and 2 was 71 minutes; average resolution time taken for grade 3 incidents was also calculated and seen to be 101 minutes.

Ideally a selected segregation length will result in a situation whereby lengths of time close to both these values of 71 and 101 minutes will be able to be used within the model. However it should also be considered that over reduction of period lengths could be detrimental to model accuracy; breaking incidents into a large amount of categories would result in a great increase to the size of the data set and thus calculation times within the model for no strong improvement to accuracy. Indeed with period lengths being particularly short there is potential to underestimate demand levels due to the historical average number of incidents occurring within these short periods being so small. Potentially this could also couple with the inaccuracy due to rounding of custodial incidents within the original profiler causing a marked unrealistic increase in the representation of demand.

Following these considerations, a segregation period length of 15 minutes was decided upon as capable of providing a suitably high resolution demand profile whilst also maintaining the required levels of accuracy and realism. Further, this value also allows time periods close to the target figures of 71 and 101 minutes to be constructed within the model; these are made through considering 5 consecutive segregation periods, totalling 75 minutes and 7 segregation periods, totalling 105 minutes respectively.

3.4.2 Influence from Recent Demand

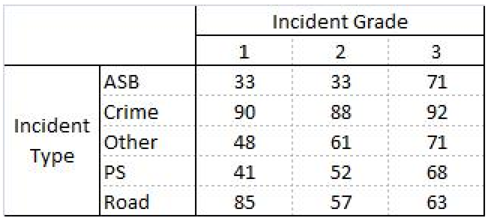
Following the adaption of the profiler and the data set upon which it is based to take account of the new segregation period length resulted in a great reduction of the overestimation of the influence from recent demand. The previous potential overestimation could reach up to 69.0%; this was reduced in the latest profiler to a maximum of 5.6%.

At this point it became of interest within the Police business to investigate the potential effects that could be borne through consideration of a break down of the recorded grade 1 and 2 incidents into different categories; each category would then have its own average resolution time for that type of incident. The 5 categories mirror the 5 major categories by which incidents are grouped together within Police records; these being anti-social behaviour, crime, public safety, road related incidents and other. Data within the profiler was then separated into each of these 5 incident types and a new series of average resolution times were calculated for each incident type and grade; the values are illustrated in Table 3.12 where the average number of minutes spent attending each of these type and grade combinations is shown. Note that these values are for only a select region of the County and are not representative of the County average of 71 and 101 minutes quoted previously. So for example, a grade 1 public safety incident in the sample region shown in Table 3.12 will be assumed to take 41 minutes to resolve; using the segregation period of 15 minutes this will round up to 45 minutes. For each time period modelled the number of active grade 1 public safety incidents will be equal to the sum of the number that start within that period and the number starting within the previous 2 segregation periods.

3.4.3 Double Crewing

There is much scope for development within the application of the double crewing factor; leading ideally to a more accurate and realistic representation of double crewing and its effects on demand. Most immediately noticeable is the previously mentioned simplified way in which the initial demand profiling tool accounts for this factor. In contrast to this approach

Table 3.12: Average incident 'assigned to released' times based on grade and incident type for a sample policing region.



		Incident Grade		
		1	2	3
Incident Type	ASB	33	33	71
	Crime	90	88	92
	Other	48	61	71
	PS	41	52	68
	Road	85	57	63

it is likely that at any time not all officers will be double crewed or single crewed and indeed a mixture of both crewing types would be expected.

Due to the nature of double crewing it is also likely that staff allocations to it will vary greatly between the differing LPUs; mostly this is because of different policing strategies to suit the different environments. As such it would be necessary either to collect and continuously update a database of suitable staffing levels or to provide capability for individual users to themselves customize double crewing levels within the model. The best approach of these two will depend on how the demand profiler is distributed by the Police force. If for example, each station receives a copy of the tool tweaked to better suit their situation then it would be of greater benefit for them to be able to adjust double crewing levels as appropriate to their needs. Alternatively the decision may be made in which the profiling tool is used to generate staffing patterns at a County or BCU level; this would require a constant flow of necessary information from individual LPUs in order to work effectively.

The double crewing factor implemented within the first profiler was very simplistic and thus in replacement for this, the final profiler makes use of an approach based upon the double crewing factor described in Section 3.2.4.3; this factor is created using Equation 3.1.

3.4.4 Directed Action Policing

The factor of directed action policing as included within the original profiler was later decided upon as not strictly necessary; mostly this being due to the fact that all such policing is planned and arrangements for suitable staff will be made without the need for profiling. Thus it was acceptable to remove this entirely from the later versions of the incident demand profiling software in order to reduce file size and thus improve speed, even if only slightly.

3.4.5 Custodial Incidents

The primary difference in the way in which custodial incidents are now represented in the latest profiler is that no rounding up takes place of the basic average arrest level data at all. Thus for any real value of such incidents occurring within a segregation period, that will be the amount of custodial incidents that are assumed to happen. In the previous version rounding up would occur for any average rate of higher than 0.09; for example then the previous version would round 0.41 incidents up to 1 incident, the new version would instead only consider 0.41 incidents. This should greatly reduce the overestimation in arrest based demand that was shown to occur in Section 3.3.1.5 and provide a much more realistic representation of this type of demand in the enhanced profiler.

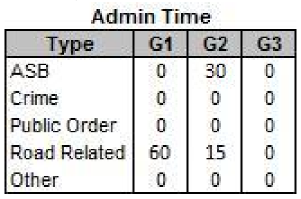
The original assumption was that 3 hours of additional time would be necessary in order to resolve any incident that resulted in an arrest. This figure was viewed by many Police staff questioned as an underestimate and conjecturally it was suggested that a value closer to 8 or 9 hours would be more appropriate. Note that with such high values it would be possible for an arrest to be ongoing past the end of a staff members' shift; in this case it would be up to a staff sergeant on duty to either authorize over time pay for the same officer to remain in attendance or arrange for the arrestee to be handed over to a new officer to continue the arrest process. Regardless of the decision made, the arrest will still require the attendance of a single officer for its duration. An average of the recorded times taken requiring the attendance of an LPO through the arrest process was calculated over all such incidents in a calendar year with the result being approximately 7 hours. In light of this an option was built in to the profiler such that a user may enter any length of additional custody time that will occur in the case of an incident resulting in an arrest; the default value of this is set to 420 minutes, or 7 hours.

3.4.6 Administrative Factor

The approach to consider administration time within the first demand profiler is very weak and succumbs to a vital flaw; namely that as staffing levels increase, so too does the amount of administrative work required to be carried out. The effects of this are especially notable when staff are set to be double crewed within the profiler; the elevated staffing level in order to tackle demand greatly increases the amount of demand due to administration. Indeed the opposite of this would be anticipated with double crewed units as the extra staff availability at each incident site would be expected to reduce the amount of time required in order to resolve any administrative work.

The enhanced profiler removes this notion of administrative time and instead allows the user the option to define figures representing additional administrative time that they estimate will be necessary on top of the average incident resolution time in order to handle administrative duties. A copy of the table used to input this information is included in Table 3.13 where entries are available for each incident type within each incident grading group (G1, G2, G3) to be allocated an integer value; this value would represent the additional time in minutes taken to resolve that category of incident. Standard administrative work required in order to resolve an incident should be included within the recorded 'assigned to released' times used in order to generate average incident durations; as such, the default values for this table within the profiler are all set to 0.

Table 3.13: Administrative time entry table used in the latest demand profiler version.



Admin Time

Type	G1	G2	G3
ASB	0	30	0
Crime	0	0	0
Public Order	0	0	0
Road Related	60	15	0
Other	0	0	0

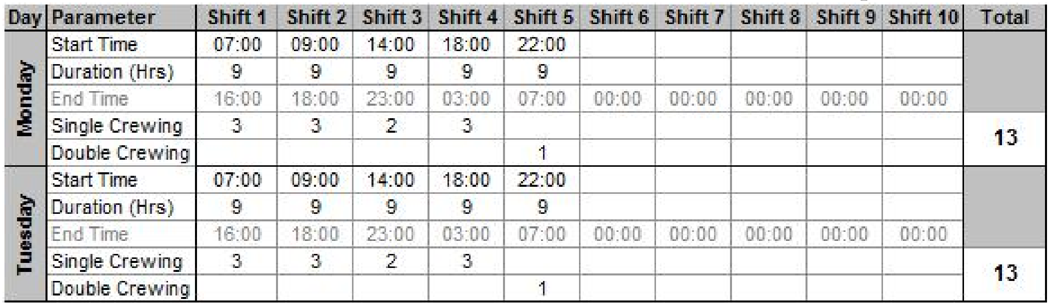
3.4.7 Abstractions

Early within the development work it was suggested that instead of considering assignation of an amount of staff and then reducing this amount to take account of abstractions, the profiler should instead provide a guide for the minimum staffing level required in order to meet all critical demands. This will mean that suggested staffing levels resulting from use of the profiler would be directly comparable to the minimum cover levels currently employed by Leicestershire Police. These minimum covers are the absolute minimum number of LPOs that must attend each shift in order to provide the minimum acceptable quality of service to the general public. Due to this it was decided to remove the effect of abstractions from the profiler and thus to now create staffing levels which could be thought of as suggestions for new minimum covers.

3.4.8 Shift Selection

To provide a much greater level of staff allocation and shift structure variation within the profiler a new system was imposed whereby a user may define the starting time, duration and staffing allocation of up to 10 shifts for each day of the week. There is no set minimum or maximum range that may be employed when creating shifts in the new profiler and indeed this is useful in allowing representation of shorter part-time shifts. This is a strong contrast to the capability of the first profiler in which 9 specific shift start times were set for each day, with the only shift length variation allowance being the use of shifts of 8, 9 or 10 hours in length. Critically with this set-up the original profiler is incapable of representing both the working hours of some part time staff members and the commonly occurring use of spare shifts within the Police organization. These spare shifts are used to staff small periods within a calendar day in order to allow for extra cover to be available for specific events or at other times when it is anticipated that there will be elevated demand on Police services. To record the desired information within the latest profiling software, the user is presented with a simple table in which they may define the details of each shift they wish to consider; an example of this table is contained within Table 3.14.

Table 3.14: Custom shift definition table within the latest demand profiler version.



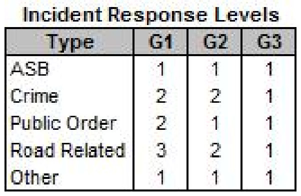
Day	Parameter	Shift 1	Shift 2	Shift 3	Shift 4	Shift 5	Shift 6	Shift 7	Shift 8	Shift 9	Shift 10	Total
Monday	Start Time	07:00	09:00	14:00	18:00	22:00						
	Duration (Hrs)	9	9	9	9	9						
	End Time	16:00	18:00	23:00	03:00	07:00	00:00	00:00	00:00	00:00	00:00	
	Single Crewing	3	3	2	3							13
	Double Crewing					1						
Tuesday	Start Time	07:00	09:00	14:00	18:00	22:00						
	Duration (Hrs)	9	9	9	9	9						
	End Time	16:00	18:00	23:00	03:00	07:00	00:00	00:00	00:00	00:00	00:00	
	Single Crewing	3	3	2	3							13
	Double Crewing					1						

The user may define the amount of single and double crewed officers assigned to each of the shifts. Spaces within the single and double crewing rows with blank values are equivalent to having 0 values, it is simply easier to immediately read staffing figures without the additional zeroes.

3.4.9 Incident Response Levels

Not all incidents will require the same level of LPO response with many types of incidents commonly requiring 2 or more Police units in attendance for safe resolution. To allow for this a new capability was introduced that will scale all incident demand by appropriate factors to simulate the varied required incident attendance levels. An example of the user input for this is included in Table 3.15; the numerical data contained within this table is purely for demonstrative purposes and is not based on real collected information. To give an example, consider that within any selected segregation period there are seen to be on average 2.71 grade 1 anti-social behaviour incidents and 1.38 grade 1 road related incidents. The multiplicative factor for grade 1 anti-social behaviours recorded in the table indicates that only 1 unit will be required to attend each of them and thus this means total demand of 2.71; the grade 1 road related incidents however would require 3 units to attend each and as such the average value is tripled giving total demand from this incident type in the segregation period of 4.14.

Table 3.15: Incident response level table used in the latest demand profiler version.



Incident Response Levels

Type	G1	G2	G3
ASB	1	1	1
Crime	2	2	1
Public Order	2	1	1
Road Related	3	2	1
Other	1	1	1

3.4.10 Graphical Representation

One common complaint with the original profiler was the complexity of the main graphical display which was deemed unintuitive for the average user. Fortunately the decision to no longer consider the effects of abstractions and instead to create a view of supply comparable to that of the currently employed minimum cover levels allowed the removal of 2 of the supply plots from the original graph; these being the total default assigned staff and the same staff minus abstractions. Indeed as the latest profiler no longer automatically distributes staff in the same way as the original it is also possible to remove the plot that was represented by this previously. Thus there only remains a single plot for the user entered staff level and a plot for the predicted demand level; this is illustrated by Figure 3.6.

On this graph the smaller red shaded region indicates the expected level of demand upon resources that would be seen over the particular day and region in question whilst the larger green shaded region demonstrates a staffing level as defined by user input shifts. This suggests that there are enough staff members available to easily be able to meet all of the expected demand, however the demand representation as shown is only the average value expected when considering all historical data. It is thus important to also consider potential variation within this demand as the average demand is only truly indicative of the maximum demand that could be statistically expected to occur within 50% of cases.

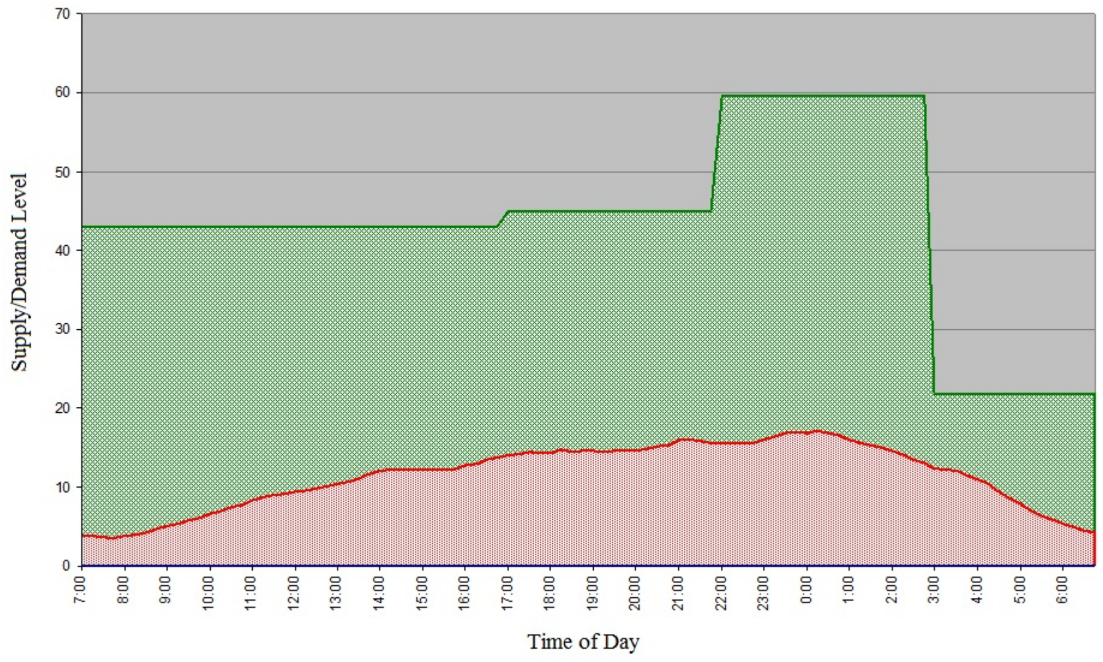


Figure 3.6: Sample graphical output of the final demand profiler software when used to simulate demand on one policing region.
The green shaded region indicates supply of staff and the red shaded region describes expected demand.

Of note however, the total expected amount of required officers is lower with the new methodology of calculation as used within this profiler in comparison to those given by the original demand profiling tool. An example of this is shown in Table 3.16 where final estimated demand values for several hours of the day within a selected policing region are shown for both the original and enhanced profilers. With a high confidence in the validity of the calculation used in order to create a profile of demand with the enhanced incident profiler it may immediately be seen that the original profiler was providing an overestimation of actual demand. Indeed for the data shown, the original profiler was predicting values on average 54.83% larger that those from the enhanced profiler. This is of key importance as it is apparent then that staffing considerations based upon the less accurate original profiler would lead to overestimation of the required staffing costs for any attempts to match staffing with demand. For example, consider only the shifts defined using the standard $2-2-2$ shift pattern employed by Leicestershire Police for full calendar week over a City region data based problem. The original profiler would suggest a requirement of 7560 staff hours to meet all demand in this case in comparison to the 6426 staff hours suggested using the enhanced profiler. As the larger value is due to a known series of overestimations, this is indicative that the original profiler could lead to unnecessary over expenditure.

Table 3.16: Estimated expected demand values for the original and enhanced incident demand profilers.

Time	17:00	18:00	19:00	20:00
Original	19.24	20.16	21.30	20.84
Enhanced	12.13	13.44	13.85	13.26

3.5 Profiler Development Stage 2. Elevated Demand Level Estimation

3.5.1 Elevated Demand Level Estimation

At this point of development, the incident profiler was capable only of evaluating the average expected level of incident based demand for each segment of the overall time period in question. It was therefore useful to simulate statistically elevated rates of incident occurrence and to evaluate their impact upon total demand levels. In order to achieve this it was first necessary to create a method by which elevated incident occurrence rates could be calculated. The underlying process is by it's nature a Poisson process and as such it would be naturally considered to institute a Poisson distribution. However in the case of the data presented within the incident profiler, non-integer values for historical incident occurrence rates are generated as input data which are further perturbed by other non-integer factors such as the double crewing factor and average incident attendance policy multiplier. Due to this it is possible to instead consider using normal approximation to the Poisson distribution.

A standard measure of the amount by which values in a series will differ from the mean is the standard deviation; alternatively the variance (being the squared standard deviation) is equally relevant. Using the calculated variances, elevated incident rates were calculated using an adjusted version of the standard z statistic hypothesis testing method[42].

To begin the adjustment, first was considered the standard form of the z statistic hypothesis test formula, namely

$$z = \frac{x - \bar{x}}{\sigma} \tag{3.2}$$

with x being the hypothesized value, $\bar{x}$ the population mean and σ the standard deviation. With knowledge of the statistical tables associated with such a test it is possible to make use of specified values of z in order to effectively reverse calculate values of x with a desired occurrence probability. This is achieved through substitution of the z value in question along with the other known values of population mean and standard deviation and through rearrangement of the equation into

$$x = z\sigma + \bar{x}. \tag{3.3}$$

The standard deviation may be calculated as usual, however in the case experienced within Leicestershire Police it was seen that this resulted in a very large calculation time as records of over 1 million incidents were manipulated using Microsoft Access. Thus it may prove of benefit to accept a slight reduction in the accuracy of these values in order to increase the speed at which a result can be provided; indeed this method was used in order to provide initial estimates of potential demand incident variation whilst calculations for accurate standard deviations were undertaken. This process is achievable through approximation to the true values for σ by consideration of the standard equation for its calculation

$$\sigma = \sqrt{\frac{\Sigma(x - \bar{x})^2}{N}} \tag{3.4}$$

and the nature of the demand particular to this problem. By this it is meant that, for each of the incident grades and types it is seen that at any individual LPU that fewer than 1 of these incidents will occur within any of the defined quarter hourly segregated time periods used within the incident profiler. The result of this is that the average number of occurring incidents within each of these categories is then between 0 and 1; importantly this means that the total number of occurrences for each time period is greater than the number of incidents seen to occur within it (each time segment will appear once for every calendar week within the time period subjected to data collection). This is of use as it means that maximum and minimum possible values for the standard deviation may be calculated.

Consider the case in which the population is comprised of N members, P of which take the value 1 and the rest the value 0, where $N > P$. This describes the situation in which deviation from the mean will be minimized; in the incident demand scenario this is equivalent to stating that the total number of incidents of the given type and grade (P) are distributed across the weeks in which they could have occurred as evenly as possible. Thus the standard equation used in calculation of σ may be rewritten into the form

$$\sigma = \sqrt{\frac{\Sigma_1^P(1 - \bar{x}^2) + \Sigma_{P+1}^N(\bar{x}^2)}{N}}, \tag{3.5}$$

which is equivalent to

$$\sigma = \sqrt{\frac{P(1-\frac{P}{N})^2}{N} + \frac{(N-P)\frac{P^2}{N^2}}{N}}. \tag{3.6}$$

A rearrangement of this equation to

$$\sigma = \sqrt{\frac{P}{N}\left(1 - \frac{2P}{N} + \frac{P^2}{N^2}\right) + \frac{P^2}{N^2} - \frac{P^3}{N^3}} \tag{3.7}$$

reveals it to be a polynomial in terms of $\frac{P}{N}$, or $\bar{x}$; accordingly then this may be rewritten for simplicity as

$$\sigma = \sqrt{\bar{x}(1-2\bar{x}+\bar{x}^2) + \bar{x}^2 - \bar{x}^3}. \tag{3.8}$$

Through collection of like terms, this provides the final result

$$\sigma = \sqrt{\bar{x}(1-\bar{x})}.$$

This final value indicates the lower bound on the standard deviation for a series of positive integer values with a known mean of between 0 and 1. In situations whereby such a lower bound is a suitable approximation of standard deviation then, use of this formula may save a significant amount of computation.

It is also possible to calculate a similar upper bound on the range of values obtainable as standard deviation for such a set of data through consideration of a different base scenario. The maximal standard deviation in such a system as above will be obtained through the situation wherein all data series values are 0 except for a single non-zero value. Thus beginning with the same base equation describing standard deviation as previously (Equation 3.4 instead now it is considered that of the population of N elements, there is a single element x taking the value of P with all others taking the value 0. This allows reformulation of the standard deviation definition into the form

$$\sigma = \sqrt{\frac{(P-\frac{P}{N})^2}{N} + \frac{(N-1)\frac{P^2}{N^2}}{N}} \tag{3.9}$$

which, through rearrangement to remove the lower denominators, becomes

$$\sigma = \sqrt{\frac{P^2}{N}\left(1-\frac{1}{N}\right)^2 + \frac{P^2}{N^2} - \frac{P^2}{N^3}}. \tag{3.10}$$

All terms contained beneath the square root are some multiple of $\frac{P^2}{N}$ and it is possible to rewrite this accordingly as

$$\sigma = \sqrt{\frac{P^2}{N}\left(\left(1-\frac{1}{N})^2\right)+\frac{1}{N}-\frac{1}{N^2}\right)}. \tag{3.11}$$

Through expansion of brackets this becomes

$$\sigma = \sqrt{\frac{P^2}{N}\left(1-\frac{2}{N}+\frac{1}{N^2}+\frac{1}{N}-\frac{1}{N^2}\right)} \tag{3.12}$$

which simplifies as

$$\sigma = \sqrt{\frac{P^2}{N}\left(1-\frac{1}{N}\right)}. \tag{3.13}$$

Indeed through simplification of the parentheses here it is possible to further rewrite this as

$$\sigma = \sqrt{\frac{P^2}{N^2}(N-1)} \tag{3.14}$$

which is equivalent to

$$\sigma = \frac{P}{N}\sqrt{N-1} \tag{3.15}$$

or, as $\frac{P}{N}$ is equal to the mean of the population $\bar{x}$

$$\sigma = \bar{x}\sqrt{N-1}. \tag{3.16}$$

With the standard deviation estimated from these new equations (which were later replaced with accurate standard deviations upon their calculation) it was possible to estimate elevated levels of incident occurrence and to visualize the effect this would have upon volume demand within the incident profiler. Considering a particular confidence level, say 90%, the particular z statistic value associated with it may be obtained using standard z-test tables, in this case the value being approximately 1.28. Inserting this value alongside the mean and standard deviation into equation (3.3) will then provide an elevated value that is an upper bound on the range of values expected in 90% of cases. The demand profiler makes use of a user specified confidence level which in turn generates a suitable value for the z statistic. In this way the base incident occurrence rates are elevated to illustrate the level of incident demand that would be expected to occur in the associated percentage of cases. Figure 3.7 illustrates an example of this for a policing region within Leicestershire with incident elevation set to a value of 80%; this meaning that all of the base incident occurrence rates are increased to signify the maximum rate of occurrence expected in 80% of cases. The graph is of the same format as others already demonstrated as output from the profiler with the addition of a thicker demand curve to illustrate the expected elevated volume of incident based demand; this being the total amount of live incidents in progress. A period of time between 03:00am and 04:30am is indicated within this graph at which the expected level of demand outstrips the supply of available resources. This clearly indicates the problem with considering only average demand; with the 80% case shown this illustrates that the Police should expect to fail to meet all demand using the input staffing roster within this period of time in at least 1 in 5 cases. The addition of this facility enables user specification of a desired percentage value, allowing planning of staff in preparation for reasonable variation in demand.

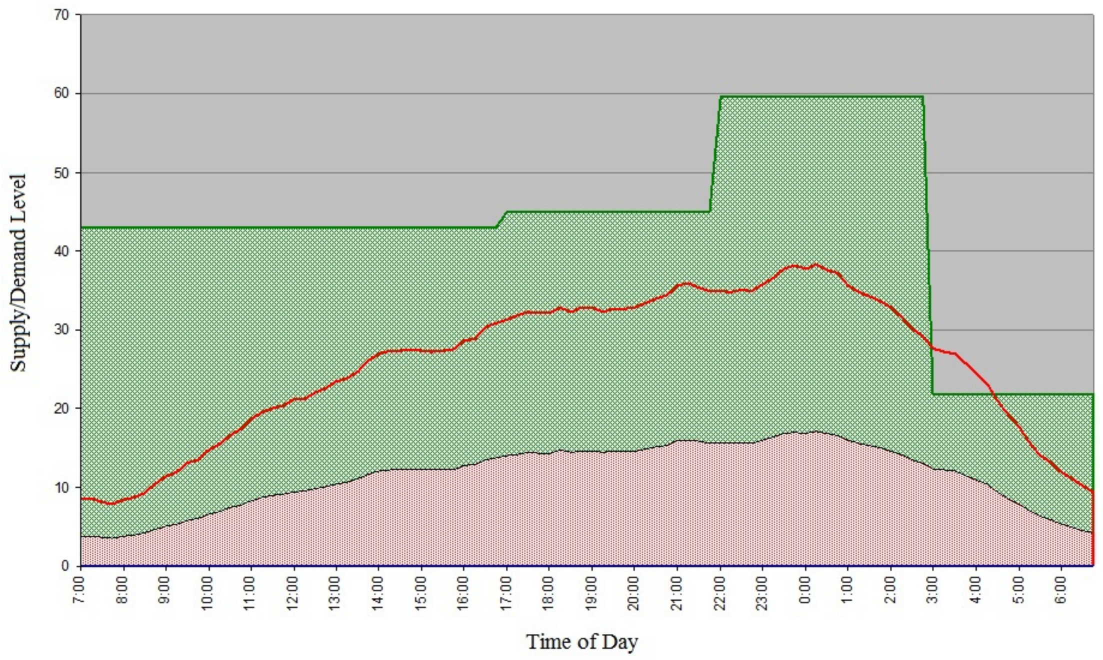


Figure 3.7: An example graph illustrating the effects of demand elevation within the incident profiler.
The green shaded region indicates supply of staff, the red shaded region describes average demand and the thick red line indicates statistically elevated demand.

3.6 Profiler Development Stage 3. Basic Simulation Method

The final piece added to the enhanced incident demand profiler was a small program that makes use of the user defined shifts in order to calculate a minimum staffing level such that all demand is met over the entire time period. This program was written using the Visual Basic components included as part of the Microsoft Excel software package. The greatest benefit this provides is the ability to read directly any data that is stored within the workbook and further to automatically take account of any changes that occur to the said data. The simulation works individually upon each day of the week and as such a loop is required such that the simulation procedure will occur 7 times accordingly. The overall process by which this method is applied is demonstrated within the flow chart within Figure 3.8

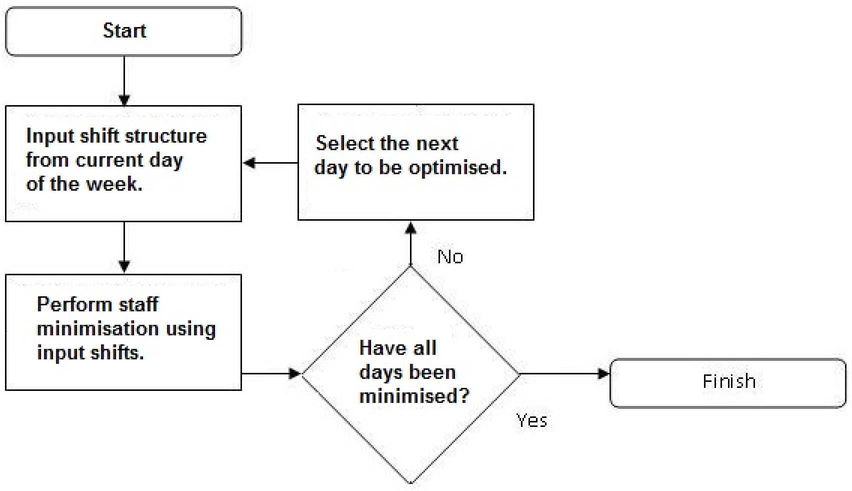
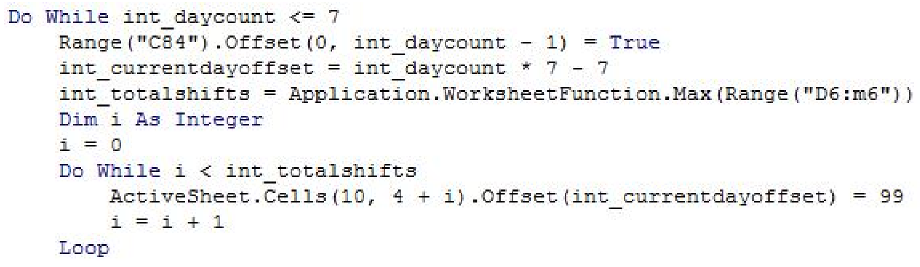

Start
Input shift structure from current day of the week.
Select the next day to be optimised.
No
Perform staff minimisation using input shifts.
Have all days been minimised?
Yes
Finish

Figure 3.8: Flow chart describing the optimization applied within the enhanced profiler.

The first stage within this loop is to initialize all of the user defined shifts to some high value; doing this ensures that later stages will find a better fit to the demand profile, and thus minimize the suggested staffing levels. This first part is achieved using the section of code displayed in Figure 3.9. The value *int_daycount* is incremented for the outer loop in order to proceed through the 7 days of the week, $int\_currentdayoffset$ defines an offset value for use in locating data relevant to the current day within the workbook and $int\_totalshifts$ calculates the total number of shifts input by the user for the current day. The smaller loop describes the process whereby each of the user defined shifts will then be assigned a relatively high value in order to initialize them; in this case the value selected is 99.

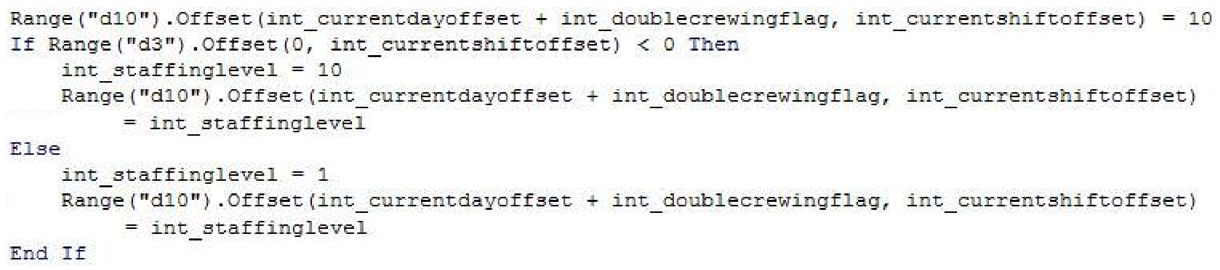
Using the value of $int\_currentdayoffset$ and a newly defined $int\_currentshiftoffset$, the



```
Do While int_daycount <= 7
    Range("C84").Offset(0, int_daycount - 1) = True
    int_currentdayoffset = int_daycount * 7 - 7
    int_totalshifts = Application.WorksheetFunction.Max(Range("D6:m6"))
    Dim i As Integer
    i = 0
    Do While i < int_totalshifts
        ActiveSheet.Cells(10, 4 + i).Offset(int_currentdayoffset) = 99
        i = i + 1
    Loop
```

Figure 3.9: Visual Basic code describing basic simulation shift value initialization.

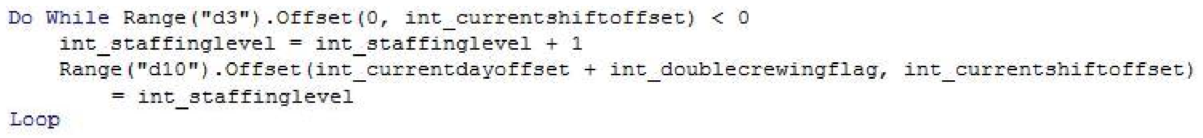
latest starting shift within the current day is selected and has the high staffing level assigned to it now reduced to a reasonable point from where the search may progress. The code used for this is shown in Figure 3.10. Here it is seen that the record used within the workbook in order to record current staffing levels for the selected shift is firstly set to the value 10, then an if statement is used to decide whether this value should remain at 10 or whether it should be reduced to 1. The reason for this is that for the next stage of the search it is important that there be some demand surplus present at some point in time over the duration of the shift, thus if assigning 10 staff will provide no such surplus then they are reduced to 1. The second line of code, initializing the if statement, makes reference to a cell within the workbook that contains the highest demand surplus for the current shift; this is then tested to see if it is negative (implying demand surplus), and if so then staffing levels remain for now at 10.



```
Range("d10").Offset(int_currentdayoffset + int_doublecrewingflag, int_currentshiftoffset) = 10
If Range("d3").Offset(0, int_currentshiftoffset) < 0 Then
    int_staffinglevel = 10
    Range("d10").Offset(int_currentdayoffset + int_doublecrewingflag, int_currentshiftoffset)
        = int_staffinglevel
Else
    int_staffinglevel = 1
    Range("d10").Offset(int_currentdayoffset + int_doublecrewingflag, int_currentshiftoffset)
        = int_staffinglevel
End If
```

Figure 3.10: Visual Basic code showing the priming of an individual shift for staff level adjustment search.

Now for the current shift, the search process will steadily increase the amount of assigned staff until all demand surplus shown within that shift has been met. This should provide the least amount of officers required for that shift to meet all of the predicted demand for its duration. The code for this final stage is displayed in Figure 3.11.



```
Do While Range("d3").Offset(0, int_currentshiftoffset) < 0
    int_staffinglevel = int_staffinglevel + 1
    Range("d10").Offset(int_currentdayoffset + int_doublecrewingflag, int_currentshiftoffset)
        = int_staffinglevel
Loop
```

Figure 3.11: Visual Basic code showing the priming of an individual shift for staff level adjustment search.

For an illustrative example, Figure 3.12 displays the results of application of this staff minimization process to the City based trial problem. Though this is able to reduce staff across defined shifts to the minimum values that are able to meet all expected demand, this does not consider any variation in shift definitions and is restricted to those created by the user.

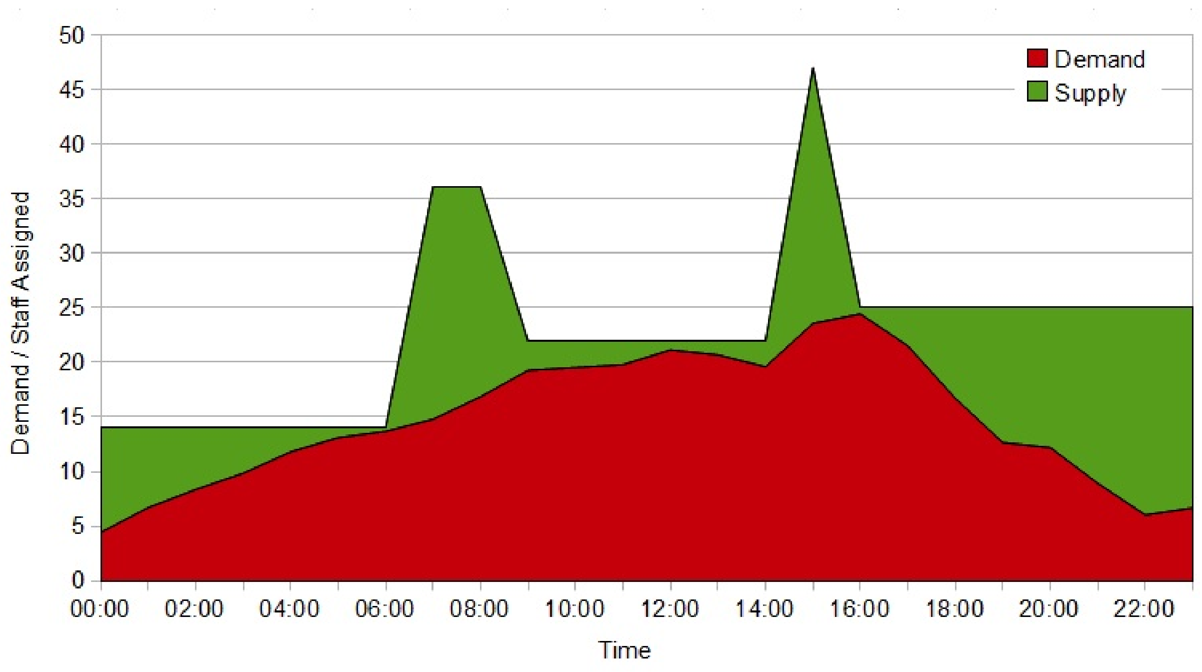


Figure 3.12: Illustrative example of results from incident profiler staff minimization. The green shaded region details the generated supply of staff and the red shaded region describes expected demand.

Overall the addition of this mechanism to the incident profiler is highly useful as it will allow a user to obtain a series of benchmark staffing levels that will meet predicted demand for any defined pattern of shifts they desire. In this way it is capable of providing at the least a starting point from which staff planners may then work in order to not only meet the predicted incident demand but also to meet other demands upon the Police service.

3.7 Summary

In this chapter a review of the demands present upon some of the the key front line departments within the Police organization have been described with particular emphasis given to any constraints and numerical observations that will be of importance for future simulation and modelling approaches.The key areas addressed in this chapter are:

- Understanding of the demands on Police officers in the Leicestershire area.
- Understanding of the current tools and method available for setting Police rosters to tackle the demand.
- Identification of the current weakness of the demand modelling technique used to enable future methods to be sought.

This chapter has presented the initial approach taken by Leicestershire Police in attempting to accurately represent the demand on their LPO staff members. Development of the original profiling software has been described through discussion of the changes made in creation of the latest incident demand profiler and the effects these changes have on countering the early profiler weaknesses. The most important changes made to the original profiler in achieving improvement were:

- The re-segregation of base incident and arrest occurrence rate data has reduced potential overestimation from rounding.
- Shifts are now definable by the user and may be set to start at any point in time and to last for any specified duration.
- Rounding of custodial incidents has been removed, much lessening the distortion of data when low demand policing regions are profiled.
- Administration time is now incident dependent and is entered by the user as additional time required for resolution of specific incident types and grades.
- Double crewing now takes account that a shift may be populated by both single and double crewed staff appropriately.
- Abstractions have been removed and the profiling software is now stated as providing results comparable to Police staff minimum cover levels.

- More data has been collected to enhance the base data set; now a period of more than 20 months is used in order to form this base.

The work of this chapter has resulted in the creation of an improved incident demand profiling tool that is able to present realistic estimates for the amount of live demand. This alone is not sufficient in order to suitably generate a staffing roster however as the profiling tool presented is only capable of staff optimization over shifts defined by a user. As such it is now of benefit to consider techniques by which shift definitions and staff distributions may be varied automatically with the aim of optimizing some objective function that quantifies the ability to meet demand. As demonstrated in Chapter 2, other researchers have noted the benefits afforded by implementation of a tabu search algorithm in tackling staff rostering problems and as such it presents a natural choice for use in such a role in this research.

Chapter 4. The Tabu Search in Detail and Basic Implementation

4.1 Introduction

This chapter will introduce and discuss one optimization method that has been applied to the demand problem described in Chapter 3, namely the tabu search, and will further describe implementation of this optimizing search algorithm to the problem through the use of a customized C++ program. A series of trials were conducted in order to both discover suitable running parameters for the search and to test potential search modifications; the results of this work will be presented in this section.

Research into the literature has shown that a number of optimization methods have been proven to offer viable solutions when applied to problems similar in structure to the allocation of LPO policing staff. Of particular note amongst these methods is the tabu search, which has been shown to be successful when used for the similar task of assigning nursing staff within Belgian hospitals [9]. The Police problem is similar in that adjustment of staff allocation is the method used in order to achieve optimization; however the objective function is greatly different. The most directly similar goal is instead to assign a specific number of staff members across a series of pre-specified shifts such that a well defined demand curve is best met; this then should suggest the best use possible of the given staff numbers and shift structure.

4.2 Initialization

For the purposes of testing developed search algorithms for their suitability in this task, a series of trial problems have been defined based on the demand profiling tool developed collaboratively with Leicestershire Police as discussed in Chapter 3. This set of 18 trial problems reflect the 3 major policing regions used within Leicestershire along with their 15 subdivisions (listed in Table 4.1) and are set to cover the period of an average calendar week. Further these problems are constructed such that the initial staff allocation is based upon currently used rosters and with a level of staff that is insufficient to meet demand over the entire time period considered. Staffing levels are selected to be insufficient in order to ensure that the optimization method will continue to perform until it finds an optimal result. In the case that far more staff than necessary are allocated, early termination of the search could occur as the algorithm would reach a point at which all demand is met and hence no more improvements could be made. This is a concern as in these cases there could be a lot of potential staffing variation that would simply be ignored due to ease of finding a first numerically suitable solution. The precise data regarding each of these trial problems are included in full within Appendix A, wherein for each problem a profile of expected demand is coupled with a suitable staffing roster.

Table 4.1: List of the 3 Basic Command Units (BCUs) and 15 Local Policing Units (LPUs) in Leicestershire.

City BCU	North BCU	South BCU
CB - Beaumont Leys	NH - Charnwood	SB - Blaby
CH - Hinckley Road	NL - Loughborough	SH - Hinckley
CK - Keyham Lane	NM - Melton	SM - Market Harborough
CM - Mansfield House	NR - Rutland	SW - Oadby & Wigston
CN - Spinney Hill Park	NW - NW Leicestershire	
CW - Welford Road		

Within these problems the objective function to be considered is defined as the sum of all demand that is not immediately met by policing staff. Figure 4.1 shows an example of a supply and demand profile, on this graph the demand not immediately being met would be that indicated in regions where the shaded (demand) area is greater than the area bounded by the thicker line (supply). Using this figure unmet demand is present between the hours from 11:00 to 14:00, 16:00 to 22:30 and 23:00 to 3:00. The score given by the objective function is the total area of the demand profile for which it rises above the thicker outlined supply region; the method of calculation for this objective function is discussed in Section 4.4.5.

4.3 Tabu Search Method

4.3.1 Introduction

Regardless of the method used in order to generate a suitably accurate demand profile, it would be of great benefit to have an automated tool capable of scheduling available staff and other applicable resources in order to aid in achieving key performance targets. One such method would be to use one of the many known optimization techniques in order to run an optimizing search procedure; of particular note due to the similar tasks for which it has previously been used is the tabu search. The tabu search is a very powerful optimization tool, the initial stages of creation and development for which are found in the text Tabu Search [8] alongside suitable accreditation for early developers. The wide range of applications for this

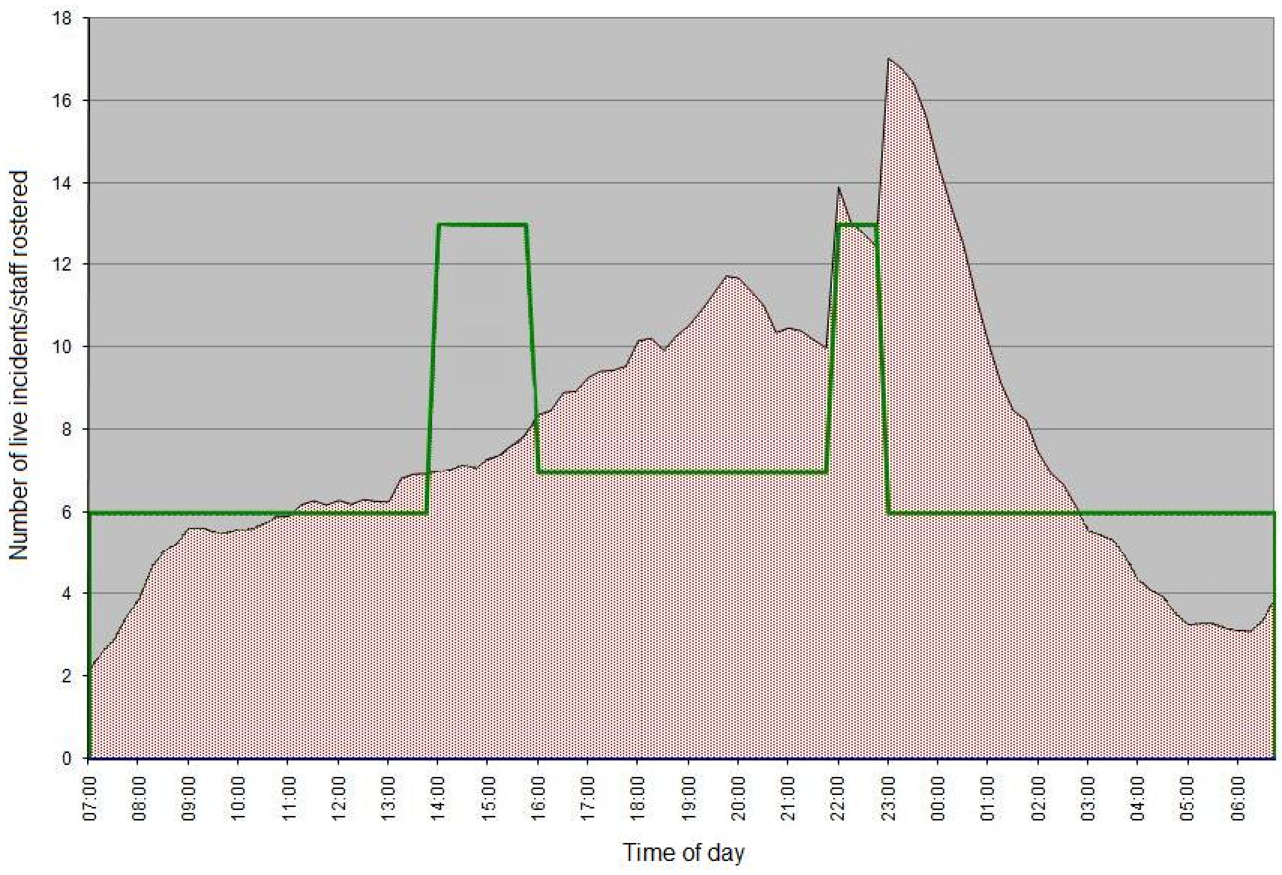


Figure 4.1: Sample demand and staffing profile graph.
The shaded region describes demand and the area encapsulated by the thicker line describes supply.

technique and its high rate of success have led to many further developments by researchers in a variety of fields. Such a wide applicability to a large variety of problems is possible mostly due to the adaptability of the method allowing a high level of customization in order to suit individual needs. For example it has been used to great effect in the problems of staff rostering [21], vehicle routing [10] and general optimization tasks [43]. In each of these the tabu search method is employed in order to solve specific optimization tasks and is shown to compare favourably against other optimizing techniques both in terms of optimization ability and calculation cost.

One of the key features of tabu search is the continuous updating of, and reference to, a flexible memory structure. The precise nature of this structure will vary with implementation yet they are all common in that they allow the consideration of effects from previous optimization attempts when deciding upon a current path for investigation.

Tabu search uses as its solution space the set of all possible solutions that exist for the given problem. It is through implementation of the search that smaller regions of this search space are investigated in order to find an optimizing solution efficiently, without having to assess each and every possible solution. The method works by first selecting an initial solution, either purely randomly or by some specific design, and then investigating nearby solutions in order to attempt to find ones that are more optimal as assessed by some objective function. Nearby solutions are defined as those belonging to the neighbourhood of the current solution, formally a neighbourhood $N(S)$ for a given solution S is the set of solutions obtained by applying a single local transformation to S. This transformation can be of many types, for example this could be by random selection of an improving move within the local neighbourhood $N(S)$ or indeed by following a more methodical approach. At each iteration of the search process then the neighbourhood $N(S)$ will be evaluated and a best improving solution S' will be found, this solution will then be adopted as the current best and have its own neighbourhood, $N(S')$, similarly evaluated. This process will repeat until some stopping criteria is reached; typically this would be after completion of a certain number of iterations or alternatively after a certain number have been completed with no improvement to solution quality. This sounds immediately very similar to local search (or steepest descent), however there are many beneficial additions made that transform this basis into the more powerful tabu search method. Namely, these are the additions of tabu tenure, aspiration criteria, search diversification and search intensification.

4.3.2 Basic Method

One of the basic concepts necessary for use of the tabu search is the idea of a 'move'. Assume that a base system has been specified for which an optimum solution, as specified by some objective function, is desired. A move is an action or local transformation by which the system is adjusted; this move usually being chosen so that the resulting modified system is considered to be more optimal. For example in a shift scheduling scenario there are a variety of possible moves, some of which could be:

Move 1	The exchange of position for two shifts with each other; for example staff member *A* will be moved from a day shift to a night shift and simultaneously staff member *B* will move from a night to a day shift.
Move 2	An adjustment of the start and end times for a shift; for example making a staff member begin and end their shift an hour earlier.
Move 3	The addition of a shift to the timetable.
Move 4	The removal of a shift from the timetable.

Consider a system where the move of type **Move 1** is implemented wherein a pre-specified group of objects are switched with each other in an attempt at optimization. As demonstrated in the work of Reeves et al. [51] it is convenient to illustrate the set of all possible moves in a case like this in a grid-like data structure; so in a system with 6 interchangeable objects this could be of the form shown in Figure 4.2. Each grid subsection here corresponds to the move which switches the position of the two relevant objects, for example the square marked with an **X** indicates the switching of objects 2 and 6.

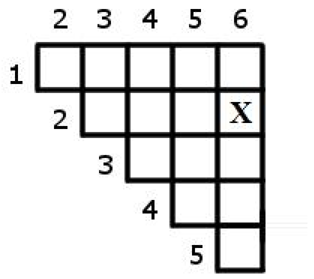


Figure 4.2: Tabu memory structure for an object switching scenario.

The idea of tabu search when applied in such a scenario is to label previously used moves as tabu, meaning that they may not be used again. So at each stage of the optimization process the most improving move will be selected and also marked as exempt from future selection. In the case of moves of type **Move 1** this will prevent the reverse move from occurring, as the selection and exchange of the same 2 elements twice will result in a return to the original state. In doing this then a string of most improving moves will be conducted until a point is reached at which no further moves will enhance the solution; at this point the result should at least be a local optimum. In this way the basic search method described can be seen as being highly similar to a steepest ascent/descent method for optimization of an objective function.

4.3.3 Tabu Tenure

In more complex cases, especially in considering different move types, when a move is designated as tabu it is assigned a 'tenure'; a value which will show how recently or frequently a particular element or combination of elements have been used. The construction and methods of application for suitable tabu tenure structures are well documented in the literature [52], [53]. From the interpretation and relevant updating of a series of tenure values it is possible to provide a measure for how soon it may be reasonable to consider using the same elements within a solution quality improving move again. The tabu tenure effectively restricts the local neighbourhood of potential search solutions at each search iteration in order to both reduce the chance of a search becoming trapped at a strong local attractor and to increase diversity of the search. There are 2 major forms of tabu tenure, recency based tenure and frequency based tenure.

4.3.3.1 Recency Based Tenure In a recency based tenure approach, a record is kept of the elements used within moves selected towards optimization and how recently their selection occurred. For example, consider the object switching example shown in Figure 4.2 and that now a tenure value is assigned to any previously conducted move; thus tenure is applied to pairs of elements and not individual elements. The tenure applied will be set to some initial value which will then decrease as subsequent moves occur, in this way the pairing with the highest tenure value will be that which occurred most recently. So with a tenure value of 3 being applied to selected moves then, after 3 moves have been conducted, the memory structure illustrated in Figure 4.3a could result. The grid presented records values that indicate how recently particular moves have been used, so for example the grid subsection indicating the switching of elements 1 and 4 contains the value 3; indicating that the exchange of elements 1 and 4 was the most recent move. With recency tenure for moves decreasing as other moves occur, the value attributed to each pairing describes how many more moves will have to be undergone before their tenure value becomes 0 and the move may be performed again without restriction in the search process. To demonstrate the effects of a further move using this same memory structure consider that after the result shown in Figure 4.3a is reached an additional move is performed switching elements 2 and 3. All tenure values within the structure would be reduced by 1 and a new tenure value of 3 would be appropriately assigned for the newly completed move; this would result in the memory structure as shown in Figure 4.3b. This has the effect of causing the tenure assigned to the move exchanging elements 2 and 6 to be reduced to 0, such that no tenure penalty for this move is considered when assessing potential future improving moves.

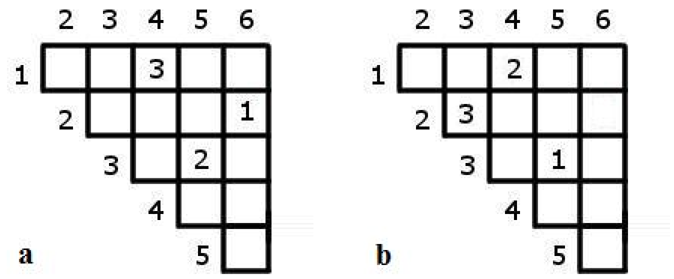


Figure 4.3: Memory structure with 3 move tenure in an object switching scenario.

The tenure values associated with previously selected search elements may be used in order to restrict the current neighbourhood of solutions in 2 different ways. The first and simplest of these is to prevent any element with a non-zero tenure value from being a candidate for solution improvement. In the example this approach will immediately prevent the reverse of previously conducted element exchanges from occurring within the next few search iterations. This method can however be too restrictive and the alternative approach is to use the tenure penalties attributed to search elements in order to penalize their quality score. In this way the chance of repeat selection is reduced however it is not completely invalidated; should a repetition of the move provide sufficient improvement to quality that it yields the best move within the local neighbourhood even despite the penalty then it may still be selected.

Selection of tabu tenure values to use within a tabu search method is often not a simple task; there are however a few key factors that are important to bear in mind when deciding upon values to trial. Small tenure values will encourage a search method to explore only small regions around locally optimizing results within the solution space. Large tenure values however will allow the search to break free from these small regions and will provide a more diverse search of the possible solutions. Thus it is sometimes considered of benefit to use a combination of both small and large tenure values, or indeed as shown by the work of Glover & Laguna [8] to instead use a dynamic tenure that is able to change at each iteration of a search. Importantly however it is hard to immediately understand for many problems precisely what would be considered a large or small tenure value. This is possible to resolve through the simple method of trial and error, investigating a range of tabu tenure values and noting their effects not only on solutions provided but also on how the search process behaves. Indeed this is only a necessary stage within the early uses and construction of the search algorithm; at the point of industrial application suitable values for wide scale use will have been discovered.

4.3.3.2 Frequency Based Tenure In addition to recency based tabu tenure, in which moves are restricted based on the most recent time of their occurrence, there is also the notion of frequency based tenure [52], [53]. The idea of this method is to record a tally of the number of times any move has been selected and use this to generate a weighting function so as to discourage their repeated use. Importantly this method allows greater variation within any search process which will be more likely to find a variety of local optima and also increase the possibility of global optimum discovery. Indeed this type of tabu tenure is only suited for use through penalization of solution quality at each search iteration.

It is usual for this tenure method to first consider each possible move and to quantify the effect that this will have upon the resulting system. This is accomplished through conducting each individual move and then assessing the resultant system state using the desired objective function, the score for each move is logged and the move is reversed. After this the frequency tenure for each move is considered and appropriate penalties are placed upon the benefit value of potential moves based upon this. In this way it is then possible that the most improving move at a given step may be ignored due to the historical frequency of selection; this will encourage the search to then progress along a previously uninvestigated path by choosing a less improving but higher rated move. This provides the benefit of potentially helping the search avoid becoming trapped at a locally optimizing result, instead making it tend towards the discovery of stronger local optima or indeed a global optimum.

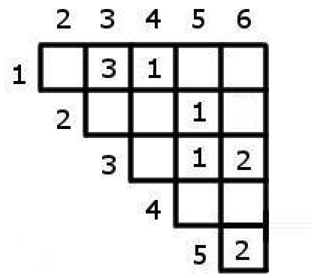


Figure 4.4: A frequency based tabu memory structure.

For example consider an object switching scenario with the frequency based memory structure as shown in Figure 4.4. In order to find the next most improving move for the system using this, firstly the relative merit of each potential move would be assessed through use of a suitable quantifying objective function. Following this the resulting scores would be adjusted using penalties derived from the tabu memory structure. The simplest method of penalization would be the direct reduction of scores using figures straight from the memory structure; for example, if the structure in Figure 4.4 were used then the move attributed with the score for switching of elements 3 and 6 would be worsened by 2. The precise way in which the penalty is calculated and applied is however very likely to vary between the different situations in which it is implemented, as clearly it will not always be the case that a direct reduction of score will be of most use. For example, consider a situation in which repeated use of the same switching is exponentially detrimental to a system as opposed to being linearly so; it would be more appropriate there to consider instead penalties to be of the value e^i where i is the current frequency of the move in question.

4.3.4 Aspiration Criteria

It is possible in some situations that it could be of benefit to consider using trial solutions or moves that would, by application of a tabu tenure, have been marked as containing tabu elements and through this considered to be unusable. In situations such that there are many long lasting recency tenures (or very large frequency tenures) it is quite possible then that the non-tabu neighbourhood solution space could become dramatically diminished leading to weaker optimization and potentially even search stagnation. Overcoming this concern is the role of aspiration criteria. These are mathematically defined methods that are able to, either temporarily or permanently, revoke the tabu status assigned to any particular move given that it meets some specifically defined criteria. An example of such a criteria is to allow a move that has otherwise been marked as invalid to be performed if it would result in a solution better than the current best known solution. This type of aspiration criteria can also be known within the literature as the aspiration threshold [8]. Certainly this is the case for aspiration based around an objective function value assessment, wherein a particular threshold value may be set such that any moves capable of improving beyond this value may be considered without penalty. A flow diagram is presented within Figure 4.5 to aid the description of a tabu search process with aspiration criteria.

First in the process a trial solution is selected, either randomly or by design, from those in the neighbourhood of the current best solution. This trial solution is then assessed as to whether or not it contains any attributes that are currently defined as being tabu-active (or tabu). In the case that no tabu attributes are present in the solution then it is evaluated as per some objective function with no additional penalties whatsoever. If however tabu attributes are present then a second test is performed, this being whether or not the solution reaches some aspiration threshold or equivalently satisfies any other type of aspiration criteria. A solution

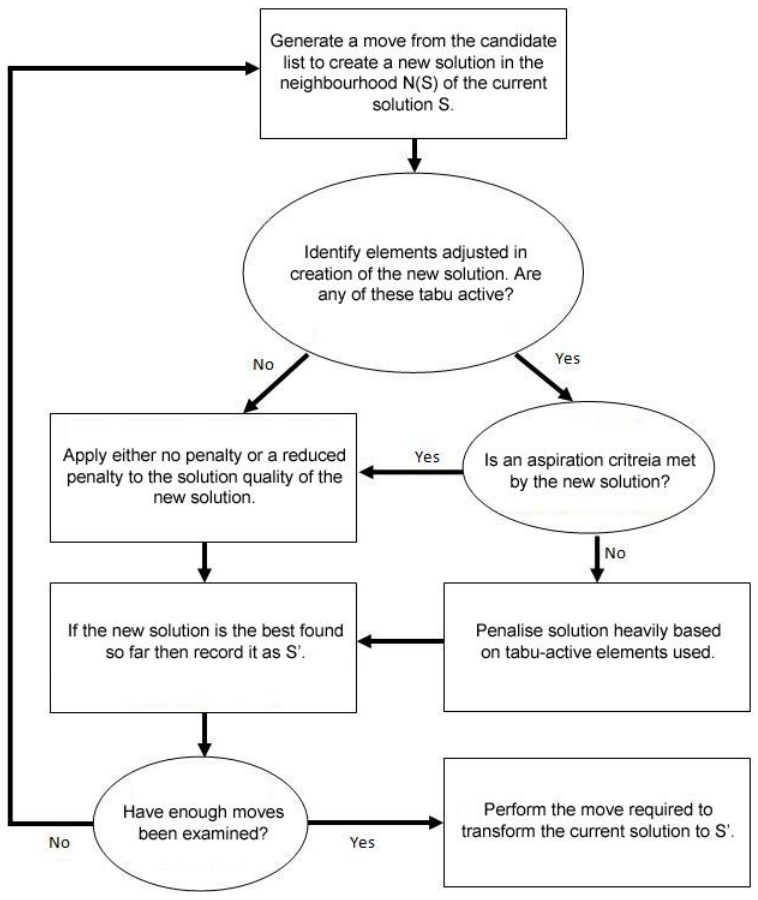

Generate a move from the candidate list to create a new solution in the neighbourhood N(S) of the current solution S.
Identify elements adjusted in creation of the new solution. Are any of these tabu active?
No
Yes
Apply either no penalty or a reduced penalty to the solution quality of the new solution.
Yes
Is an aspiration critreia met by the new solution?
No
If the new solution is the best found so far then record it as S'.
Penalise solution heavily based on tabu-active elements used.
Have enough moves been examined?
No
Yes
Perform the move required to transform the current solution to S'.

Figure 4.5: Flow diagram showing the tabu search process with aspiration criteria.

containing tabu attributes but also reaching the aspiration threshold will be assigned no penalty and will undergo the standard objective function assessment. Solutions that contain tabu attributes and also fail to meet the aspiration threshold will be subject to the same objective function assessment but with the addition of penalties based on the number and scale of tabu attributes present in the solution. At this point in the process the solution will have been allocated a score representing its quality that may be compared against the scores of other trial solutions. In the case that the new trial score, that may include penalties, is better than the best previous one then the new trial move will be marked as the most improving. The final step of the search process then is to decide whether or not enough trial solutions have been assessed in order to be satisfied in performing the best of them. This process would then be repeated until a specified stopping condition is met and a final best solution is provided.

4.3.5 Diversification and Intensification

Aspiration criteria alone may not be enough to prevent the stagnation and possible cycling of a search procedure; in these cases it is of benefit to consider the use of diversification strategies. These are mechanisms which encourage a much larger area of the search space to be investigated; this complements well the standard tabu search which can become prone to searching only a local region of the space.

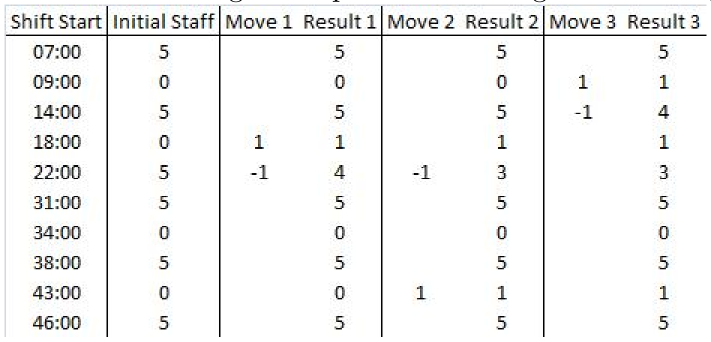
In order to achieve diversification the search must be in some way forced into a very different region of the search space. This is usually accomplished through application of one of the major diversification techniques, namely restart diversification or continuous diversification. The first of these methods, restart diversification, is used by selecting a few of the variable parameters and, upon completion of a full search, forcing them to take infrequently encountered values. These variable parameters would be any part of the solution that may be adjusted using any of the search transformations (or moves) defined for use by the search procedure. For example, in the case of a Police staff roster optimizing search a final solution of the form shown in Table 4.2 could be provided as a result of search application.

Table 4.2: Sample staffing roster for restart diversification example.
Note that shift start values between 24:00 and 48:00 indicate time within a second day and values between 48:00 and 72:00 a third.

Shift Start	Shift Length	Staff Level
07:00	9	5
09:00	10	2
14:00	9	4
18:00	9	1
22:00	9	3
31:00	9	4
34:00	10	4
38:00	9	3
43:00	9	2
46:00	9	5
55:00	9	5
57:00	10	1
60:00	9	3
64:00	9	0
70:00	9	3

In reaching this stage a number of iterations would be anticipated, each describing a different roster on the path to optimization. Within all of these rosters some combinations of shift start times and staffing levels will be encountered less than others, these make ideal candidates for restart diversification; they have been shown to provide some benefit towards optimization, however they are not the most commonly used. Table 4.3 demonstrates the first 3 moves towards optimization that were taken in reaching this result. Within this figure the shifts that donate and receive staff at each iteration are noted (marked in the figure by the '1' and '-1' notations) and the roster resulting from each change in staff distribution is shown. It is possible through use of such records to maintain a tally for each shift of the number of times it is used as either a donator of receiver of additional staff. Upon selecting suitable variables to fix, the search is restarted with these newly defined values being taken as initial conditions; this will force the search to begin in a region that was previously not investigated very thoroughly. This does lead to the issue of the stopping condition that should be placed upon the use of this diversification step; indeed through continuous use the entire solution space could be investigated, though this would be expected to be extremely inefficient and time consuming. Thus a common approach would be to limit this to only a few applications of the diversification; the precise number would be based on the expected size of the solution space. For example, in a problem with a particularly large solution space it would be considered that either a larger number of individual diversification stages or use of more powerful diversification would be necessary to suitably explore it. With each additional application of this diversification the confidence in the best solution as being optimal would increase. A good stopping condition could thus be the event of having conducted a certain number of diversifications consecutively each of which results in no improvements to the current best solution found.

Table 4.3: Table of results showing development of staffing roster through search stages.



Shift Start	Initial Staff	Move 1	Result 1	Move 2	Result 2	Move 3	Result 3
07:00	5		5		5		5
09:00	0		0		0	1	1
14:00	5		5		5	-1	4
18:00	0	1	1		1		1
22:00	5	-1	4	-1	3		3
31:00	5		5		5		5
34:00	0		0		0		0
38:00	5		5		5		5
43:00	0		0	1	1		1
46:00	5		5		5		5

The alternative major approach, continuous diversification, is applied throughout as part of the search procedure itself. This is accomplished through applying a penalty to scores obtained based on the frequency with which specific elements of the trial solution have appeared in previous solutions. These penalties should be set to small values as otherwise many potentially substantially improving moves could be ignored. The effect is a subtle one that encourages the search to make more frequent use of all available moves/elements when creating trial solutions; this is very different to the large changes forced as part of restart diversification.

In contrast, intensification instead encourages the continuation of a search within a specific local region. To do this the values of system variables (or the presence of system elements) is monitored continuously as the search procedure is conducted. After the same variables are noted as being present in a certain number of successive current best solutions the intensification process is initialized. These variables that are noted to have been present in the previous best solutions are then fixed such that they will be common members of all future solutions. The search process will then continue but will only be able to stray into the region of the search space in which solutions possess all of these common elements. Effectively then this process is an attempt to identify key patterns within solutions which are beneficial to the solution quality and then further to find the best solution available that uses this pattern.

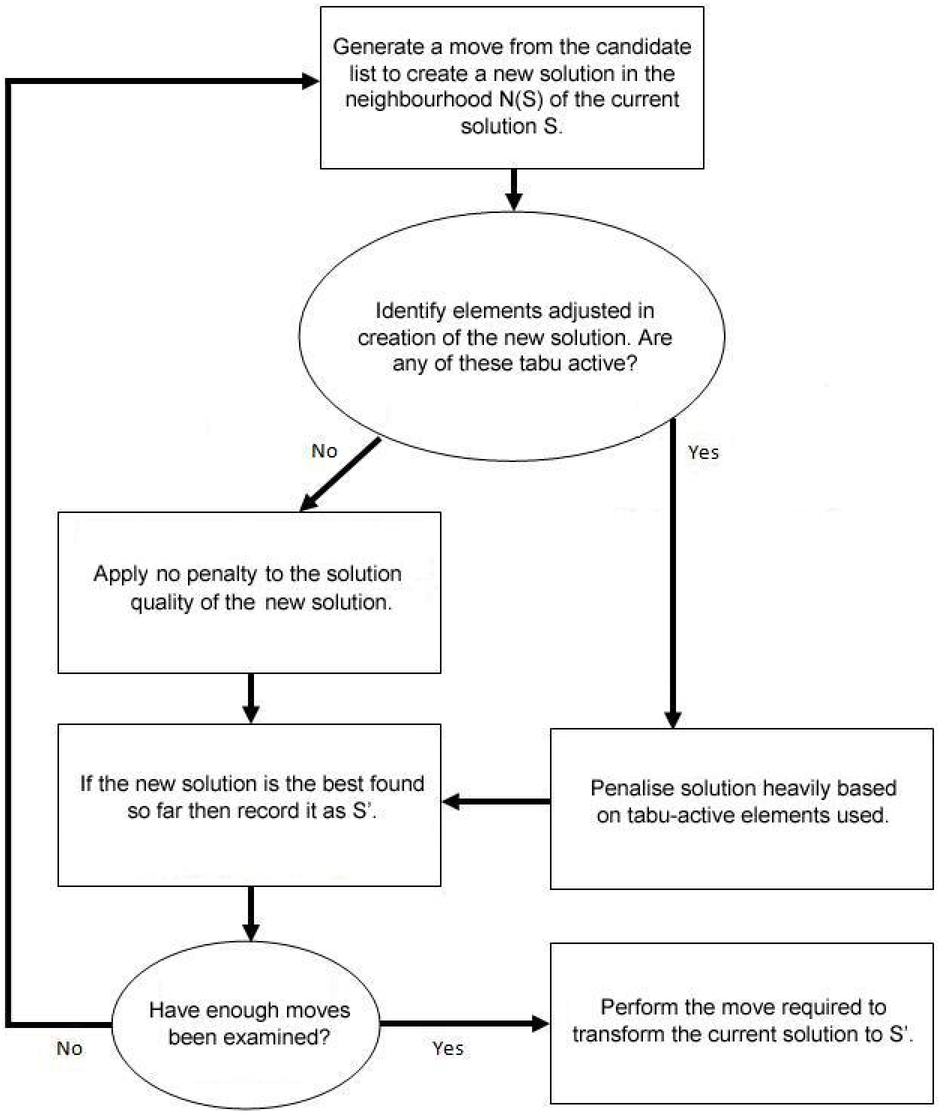
4.3.6 Stopping Criteria

The search will continue through iterations of trial solution investigation and moving between neighbouring best improving solutions until some specified stopping condition is reached. Fairly commonly used is the condition that a specific number of iterations has passed, [12]; this number should be, by design, sufficiently large so as to be confident that a wide enough search of the solution space has been performed in providing a strong optimizing result. However this approach will not guarantee in all cases that a minimum is discovered and as such an alternative method is often adopted. This alternative method is to continue performing the search until a set number of iterations have passed with no improvement made to the best found solution. In addition to definitely providing a minimal solution, with a large enough number of iterations required before stopping, there can be good confidence in the final result as being globally optimal.

4.4 A Tabu Search Algorithm for use in the Police LPO Staffing Problem

4.4.1 Overview

The tabu search process can be summarized as an optimizing search that follows a routine similar to that of a local neighbourhood (or steepest descent) search. It differs in that it considers past actions of the search in penalizing certain potential search options at each iteration. The method by which a basic tabu search acts can be thought of as that shown in the flow chart of Figure 4.6.



Generate a move from the candidate list to create a new solution in the neighbourhood N(S) of the current solution S.
Identify elements adjusted in creation of the new solution. Are any of these tabu active?
No
Yes
Apply no penalty to the solution quality of the new solution.
If the new solution is the best found so far then record it as S'.
Penalise solution heavily based on tabu-active elements used.
Have enough moves been examined?
No
Yes
Perform the move required to transform the current solution to S'.

Figure 4.6: Flow chart describing basic tabu search operation.

4.4.2 Tabu Moves

A program has been written using C++ in order to try implementation of the tabu search method upon the defined series of test problems (described in subsection 4.1). In order to achieve optimization this program allows for 3 different moves to be conducted that result in adjustment to staffing levels contained within a roster. Thus, given a starting roster this program will attempt to use these moves in order to reallocate the available staff in the best possible way. The 3 moves used are:

- The movement of a single staff member from a particular shift within the roster onto a new shift. This provides a direct method of adjusting the current staffing level and redistributing it for a better fit to the demand curve.
- The increase of the starting and finishing time of a shift to a time later in the roster. Shift length will be maintained through this move as the ending time for any adjusted shift will be increased by the same amount as its start. This provides scope to investigate the potential benefits from adjustment of the shift structure itself in addition to the allocation of staff. From this then not only will the best fit of staff be found but also the best underlying shift pattern. Note that the first occurring shift is set to be exempt from this move in order to help ensure that no roster may be created in which a period of time exists at which it is impossible to assign staff.
- The decrease of the starting time of a shift. Though very similar in nature to the previously described start time increase move, it was found more appropriate to include this as a separately available action; this being due to a resulting decrease in complexity of the program whilst still providing the same capability. This coupled with the possible increase in shift start times will allow maximum flexibility in shift starts.

Initially the program was designed to perform one of these 3 types of move until an optimizing solution is found, whereupon it will then perform another type of move to attempt further optimization. The order in which move types are selected is through manual design by the user; for example the user could elect to perform the first type of move until a local optimum is found and then to follow this by performing the second type of move. This would be repeated until no further improvements could be made to the roster at which point the program would terminate. Indeed labelling the 3 types of move from 1 to 3 as they appear in the list, it is possible to construct a string that would describe the search process. For example the string $1, 3, 2$ would correspond to a search in which staff are firstly redistributed according to the first type of move until an optimal result is found. Following this the third type of move would be used trialling the decrease of starting and ending times of shifts, again until finding a minimum whereupon the second type of move (increasing shift start and end times) is used.

4.4.3 Tabu Tenures

A critical element to the success or failure of a tabu search procedure is setting the value of tabu tenures and considering how they are to affect the search. In this work a tabu tenure penalty is applied to any elements involved within any move chosen to transform a current solution to an improving neighbour. These penalties are fixed to some values and initially it is not known for this problem precisely what values would be best suited; there is also little help to be obtained from the literature where often no mention of the exact tenure values for problems similar to this one are given. Thus a series of searches were conducted in which the tabu tenure values themselves were varied through a range of possible values in order to attempt to discern a suitable range for general use in the specific LPO staff demand profiling problem. Each move type was assigned an individual tabu tenure penalty for application to elements involved upon selection of an improving move. The final results and details of this tenure variation testing is discussed within Section 4.6.

These tabu penalties form directly the penalties that will be applied to roster scores where appropriate throughout the tabu search procedure. As minimization of demand surplus is the desired objective of optimization, the penalties will be added on top of the surplus value in order to increase the associated score of a solution and thus make solutions using tabu-active elements less desirable. So, for any trial solution considered that contains tabu-active elements, the current tabu tenure values for each of the tabu-active elements will be added cumulatively to the surplus demand. A solution containing many tabu-active elements will then be penalized more heavily than one containing fewer or no such elements. As future iterations of the search occur the tabu tenures that have been assigned to elements will be reduced until the element may be used once more without penalty.

4.4.4 Inputs and Outputs

As input the program uses a pair of .txt format data files that describe a demand curve and a user defined staffing roster each covering the same period of time. The demand curve input data file simply contains a series of values arranged in ascending chronological order that give a numeric representation of demand. The data for this demand profile may be drawn from 2 primary sources. The first of these is a piece of software, named ProfInc, employed by Leicestershire Police which attempts to suggest future demand levels based on historical information drawn from a continually updated database; indeed the development of this piece of software has been part of this research and is documented in Chapter 3. The second demand profile source is in the form of a C++ program that uses the same historical data to probabilistically construct a pseudo random profile. Table 4.4 describes the numerical information stored in a sample demand data .txt file, where 'demand' is the average number of recorded incidents in the associated hour; when coupled with the staffing data as illustrated in Table 4.5 this can provide a profile of overall demand and staffing for a preselected time period and Policing region. Indeed the incident profiling software contains data on incident demand that may be fully broken down by incident grade and type. This provides the potential to create trial problems for the tabu search software which, through their solution, could yield staffing numbers allowing for teams of personnel dedicated to specific demand types. The probabilistic C++ model in making use of the same base data source as ProfInc is equally capable of such flexibility in demand profile construction.

Table 4.4: Sample demand levels by time of day.

Hour	Demand	Hour	Demand	Hour	Demand
7 : 00	5	15 : 00	7	23 : 00	22
8 : 00	6	16 : 00	10	00 : 00	19
9 : 00	11	17 : 00	8	01 : 00	21
10 : 00	11	18 : 00	9	02 : 00	22
11 : 00	8	19 : 00	15	03 : 00	19
12 : 00	4	20 : 00	16	04 : 00	18
13 : 00	4	21 : 00	12	05 : 00	17
14 : 00	4	22 : 00	18	06 : 00	20

An input staffing roster contains a tabulated set of values similar to that described in Table 4.5 wherein a series of shifts are defined with a start time, length and initial staffing level. For the purposes of this work the standard time unit used is the hour and as such shifts are set to start at the beginning of the specified hour and run for some whole amount of hours before ending; further, individual demand values will be that for a full hour. These 2 primary data inputs are stored within a pair of .txt files as opposed to being within a single such file in order to allow the user to more easily edit or interchange various staffing rosters whilst maintaining the same demand levels.

There are 2 outputs that the program is currently designed to create if specified by the

Table 4.5: Example standard minimum cover shift pattern.

Shift Start	Shift Length	Initial Staff
7 : 00	9	6
9 : 00	9	0
14 : 00	10	7
18 : 00	9	0
22 : 00	9	6

user. The first of these is an output roster, of format identical to the input one, describing the best roster found for the given problem and its associated score. This roster will show the end result of the optimizing search procedure and describes an optimum distribution of the initially available staff numbers. Table 4.6 provides an example output roster after application of the improving search procedure. Parameters that have been varied through application of the optimizing search are the 'Staff Amount' and 'Shift Start' values; however in this example it has only been beneficial to redistribute available staff around the already defined shifts.

Table 4.6: A sample output roster resulting from the optimizing search method when applied to a trial problem.

Shift Start	Shift Length	Staff Amount
7 : 00	9	2
9 : 00	9	4
14 : 00	10	3
18 : 00	9	6
22 : 00	9	4

The second major default output is a matrix of scores from the best rosters found through variation of key search and system parameters. This provides a collection of scores that represent the ability of a set of calculated rosters in successfully meeting the preset demand profile. Different members of this set will be constructed through usage of the search process with different system parameters; of primary interest in this work were the tabu penalty values assigned for moves conducted. The scores themselves are quantified by use of some suitable objective function. The first such function used being defined as the sum of differences between supply and demand of resources for each point of time over the period in question at which demand is not met; thus they represent the total amount of demand that will not be immediately met by the associated staff roster. This objective function is discussed in detail in the following section.

4.4.5 Objective Function

In order to be able to assess the quality of any given solution, with regards to its ability to meet demand, an objective function is used. This objective function quantifies into a single real value the amount of demand that is not immediately met for a given solution of the problem being considered. To do this the difference between supply and demand of resources is considered for each individual point of time in turn within the entire time period for the problem. If the demand for resources at any point is greater than the supply then the difference between the two is added to a running score total. If instead the supply of resources is equal to or greater than the demand level then no change is made to the score; at these points in time there is no surplus demand to be added on. The demand surplus as understood by this objective function is illustrated graphically within Figure 4.7.

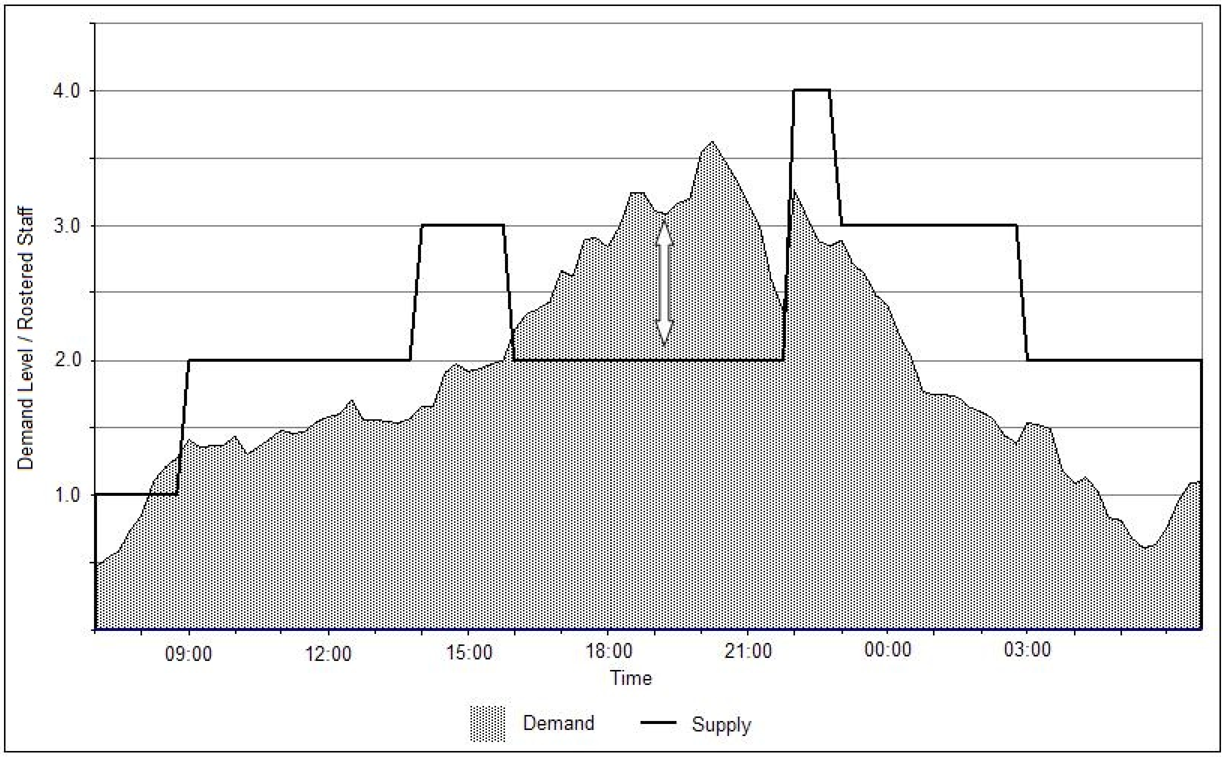


Figure 4.7: Sample resource supply and demand graph with surplus demand region indicated. The shaded region indicates expected demand whilst the thick black line describes supply of staff.

The arrow on this figure indicates an area in which demand is higher than supply, indeed performing the running total sum as previously described should provide the total area of all such regions. This can be expressed by the formula

$$\sum_{t=7}^{174} D_t - S_t \quad \forall S_t < D_t.$$

Where t ranging from 7 to 174 is used to identify the hours within a calendar week, starting at 7 to illustrate that the first hour is 7am; in the general case t could be taken to vary between suitable values that cover the entire time period represented by the demand profile in question. D_t and S_t are the demand upon resources and the supply of resources at time t respectively. Figure 4.7 describes data used in the creation of the plot shown in Table 4.7 for the time period between 15:00 and 17:00. For each of the quarter hourly data points in this example the demand for resources is greater than the total allocated supply, thus the total objective function value for this 2 hour period will be equal to the total supply minus the total demand; $23.19 - 16$ which is 7.19.

Table 4.7: Sample demand and supply values to illustrate calculation of objective function.

	15:00	15:14	15:30	15:45	16:00	16:15	16:30	16:45
Demand	2.68	2.63	2.77	2.89	2.81	2.96	3.20	3.25
Supply	2	2	2	2	2	2	2	2

Thus it would be expected that an optimizing solution would be one that provides the best fit of resources to the demand curve as possible; in effect this would be equivalent to minimization of this sum. However the trial problems have been created so that meeting all demand is not possible as this will force the search to continue attempting optimization until a stopping criteria is met. If meeting of all demand was possible then effectively this would mean that the objective function could reach a value of 0; this would then necessitate the creation of a secondary objective function in order to allow the search to continue. The benefit of doing this could be to begin further optimization through the reduction of staffing levels where possible to do so whilst still meeting all demand.

4.4.6 Stopping Criteria

Within the first search algorithm presented in this work a different stopping criteria to both of those mentioned in Section 4.3.6 is used; though there is some similarity to the latter of those previously mentioned. As a reminder, the former of those criteria was for the search to terminate after a specified number of search iterations and the latter for the search to terminate after an amount of iterations with no improvement to solution quality. The reason for this was to first create a simple stopping criteria in order to allow the search function to be tested at a basic level before more complex stopping criteria and diversifications are considered. Thus the results from this will provide a benchmark figure from which other more complex techniques may then be assessed. First the search algorithm will proceed until no more improving moves can be found in the neighbourhood surrounding the current best solution. At this point the current best solution will be perturbed and the search will then continue. In order to perturb the best solution the 5 most recently occurring moves will be reversed in order to force the search back along the search path to a previously considered solution. However whilst the most recent moves are essentially undone their tabu tenures will be retained. With the highest tenure being 5 this illustrates the primary reason why 5 moves are reversed; in reversing this number it is assured that each reversed move will still be penalized to some degree by tabu tenure, discouraging its repeat selection. This will discourage the search from making the same choices of move and providing the same best solution again due to the heavy score penalties that would be involved in reusing the still tabu-active elements. This process of obtaining an optimizing result, back-tracking from it and then continuing the search is repeated 9 times so that in total 10 locally optimizing solutions are found; the best of these being selected as the most optimal result. This figure of 10 local optima was settled upon after a series of trial optimizing searches allowing up to 1000 local optima showed the latest discovery of a solution improving upon the best found at the 10^{th} unique optimum found.

4.5 Implementation

4.5.1 Overview

The overall process by which the C++ based tabu search procedure is conducted may be summarized by the flow diagram represented in Figure 4.8. The individual stages that comprise this process will be discussed in detail within this section. Firstly the search needs to initialize through the input of some starting roster from an external data source, which in this case is in the form of .txt based files. After this the tabu search process will be conducted and will investigate the variation in staffing levels across all of the defined shifts. Following this investigation into variation of the shift starting times themselves will occur with further attempts at staffing redistribution across newly defined shifts being performed to ensure their best use.

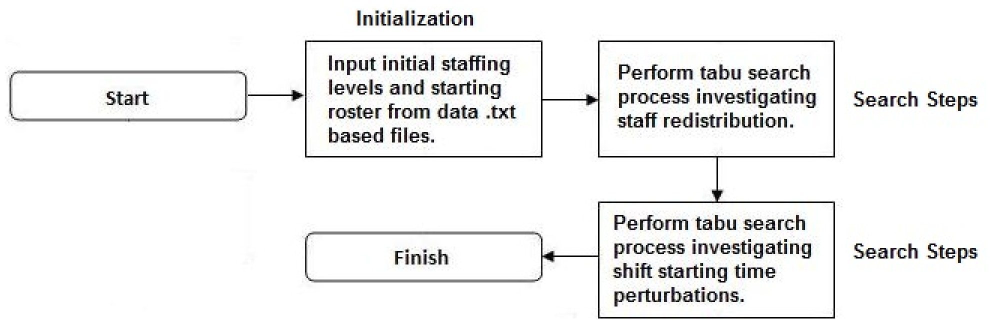
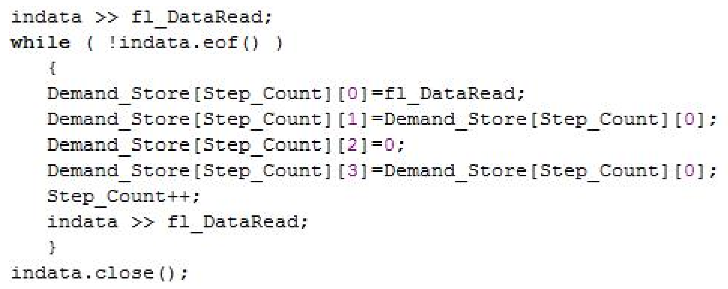

Initialization
Start
Input initial staffing levels and starting roster from data .txt based files.
Perform tabu search process investigating staff redistribution.
Search Steps
Perform tabu search process investigating shift starting time perturbations.
Search Steps
Finish

Figure 4.8: A flow chart describing the overall tabu search process implemented using C++.

4.5.2 Initialization

The very first stage is to input data from the relevant .txt files and in doing so setup the problem which is intended for optimization by the search process; this results in the storage of input data in a series of real and integer value arrays. This is performed within the C++ code by use of a coupled 'Preset_Data_Input' and 'Setup' routine. The first of these 'Preset_Data_Input' is involved with the reading of data from the input .txt documents and the updating of relevant arrays. An example of this for the input of demand data is displayed in Figure 4.9. This code initiates by storing the first value stored within the input as 'fl_DataRead'. A while loop is then implemented that will continue to perform for so long as the most recently input data value is not equal to the end of file marker; this being an automatic marker that does not require any level of additional coding. Whilst the end of file is not reached the input data value is used in this case to update a series of arrays. The first of these 'Demand_Store[][0]' records the input value itself as an unperturbed backup value for later use in restoring conducted tabu searches to the initial state without having to input the data file again. 'Demand_Store[][1] is used to record the current level of demand that remains unmet and is perturbed as the search proceeds to take account of staff members assigned; 'Demand_Store[][2]' similarly records the amount of met demand and is also perturbed to account for staff members added. 'Demand_Store[][3]' is a replica of 'Demand_Store[][1]' which is used to allow for comparison of objective score of a current best roster with a newly defined trial roster.

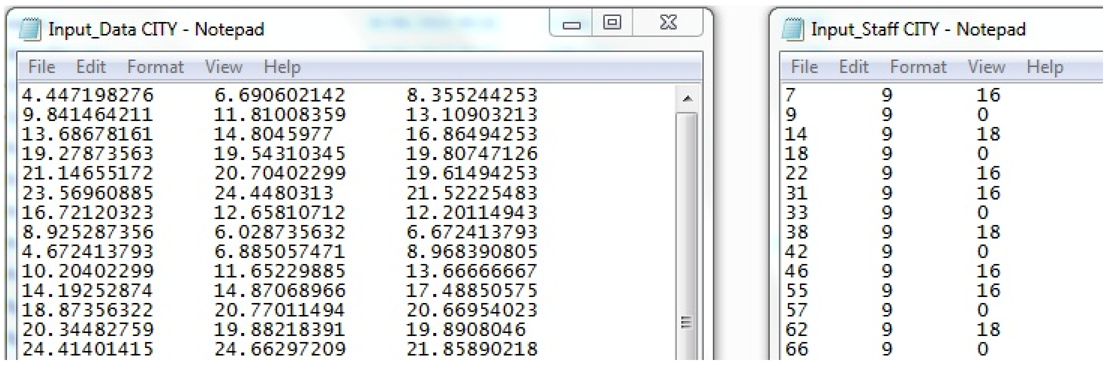
Similar code is used in order to input the initial structure of defined shifts with associated staff members. Shift starting times are recorded in a 'Shift_Start[]' array, shift durations in a 'Shift_Length[]' array and initial staffing levels in a 'Shift_Staff[]' array. Each of these elements relate directly to each other in that the N^{th} elements of each array will combine to fully describe the structure of the Nth shift. For example shift 3 would have start time, length



```
indata >> fl_DataRead;
while ( !indata.eof() )
   {
   Demand_Store[Step_Count][0]=fl_DataRead;
   Demand_Store[Step_Count][1]=Demand_Store[Step_Count][0];
   Demand_Store[Step_Count][2]=0;
   Demand_Store[Step_Count][3]=Demand_Store[Step_Count][0];
   Step_Count++;
   indata >> fl_DataRead;
   }
indata.close();
```

Figure 4.9: Code used to input data from .txt files using C++.

and initial staff described by Shift_Start[3], Shift_Length[3] and Shift_Staff[3] respectively. Examples of the structures of both the demand data and staffing data input files are included within Figure 4.10.


Input_Data CITY - Notepad
File Edit Format View Help
Input_Staff CITY - Notepad
File Edit Format View Help

```
4.447198276     6.690602142     8.355244253
9.841464211     11.81008359     13.10903213
13.68678161     14.8045977      16.86494253
19.27873563     19.54310345     19.80747126
21.14655172     20.70402299     19.61494253
23.56960885     24.4480313      21.52225483
16.72120323     12.65810712     12.20114943
8.925287356     6.028735632     6.672413793
4.672413793     6.885057471     8.968390805
10.20402299     11.65229885     13.66666667
14.19252874     14.87068966     17.48850575
18.87356322     20.77011494     20.66954023
20.34482759     19.88218391     19.8908046
24.41401415     24.66297209     21.85890218
```

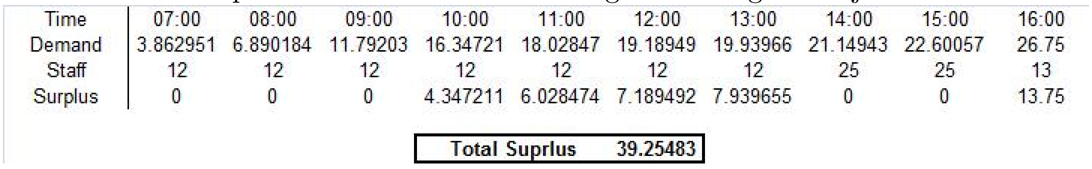
```
7       9       16
9       9       0
14      9       18
18      9       0
22      9       16
31      9       16
33      9       0
38      9       18
42      9       0
46      9       16
55      9       16
57      9       0
62      9       18
66      9       0
```

Figure 4.10: Sample input data files for the tabu search algorithm.

Having constructed an initial setup for the search it is important to assess it in order to obtain a benchmark value recording the original score attributed to the roster; without this both the effects of the search and indeed whether optimization were actually occurring at all would be unknown. This score is calculated as the total amount of demand defined using the input demand profile that is not met by the initial staffing roster. As an example Table 4.8 contains sample staffing and demand data based upon that for the hours between 07:00 and 16:00 in the City region of Leicestershire on a Monday.

For each time in the table the number of staff available on duty and the amount of demand expected to occur within that hour are recorded. Demand is drawn from the incident profiler software discussed in Section 3.9. This will calculate demand per time unit of interest using

Table 4.8: Table of sample initial demand and staffing data alongside objective function example.



Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00
Demand	3.862951	6.890184	11.79203	16.34721	18.02847	19.18949	19.93966	21.14943	22.60057	26.75
Staff	12	12	12	12	12	12	12	25	25	13
Surplus	0	0	0	4.347211	6.028474	7.189492	7.939655	0	0	13.75

Total Suprlus	39.25483

a formula equivalent to

$$D = (I + A)d,$$

where D represents total demand, I the amount of live incidents requiring attendance within the time period, A the number of arrests in the same time period and d a factor for average number of units attending an incident.

To find the total surplus demand, first the surplus for each individual hour is calculated; note that negative surplus are not recorded and are instead shown as zero values in order to consider only the times at which demand is not met. For the entire time period being considered these individual hourly surplus values are then summed into a total surplus demand figure that is equivalent to the score of the roster for that time period. In later stages when tabu tenure penalties are considered it will be this score that is adjusted where necessary to take account of these penalties.

A 'Score' routine is used in the C++ program in order to calculate this base objective function value, the bulk of the code for which is shown in figure 4.11. 3 different 'Pattern_Score[]' values are calculated within this routine with 'Pattern_Score[0]' compiling the sum of all demand, 'Pattern_Score[1]' the sum of all unmet demand and 'Pattern_Score[2]' the sum of all met demand; of these 'Pattern_Score[1]' is the target objective value. Importantly the routine initiates by setting all of these scores to 0 allowing repeated calling of this routine throughout the search as a whole; indeed this 'Score' routine is used to calculate the relative strengths of each new trial move proposed as the tabu search progresses. So for example, with the values indicated in Figure 4.8 the objective function value would be equal to the sum of all surplus demand values shown; in this case this is $4.347211 + 6.028474 + 7.189492 + 7.939655 + 13.75$ which is 39.25483.

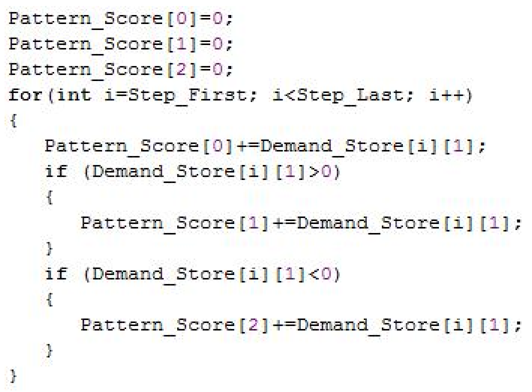

```
Pattern_Score[0]=0;
Pattern_Score[1]=0;
Pattern_Score[2]=0;
for(int i=Step_First; i<Step_Last; i++)
{
   Pattern_Score[0]+=Demand_Store[i][1];
   if (Demand_Store[i][1]>0)
   {
      Pattern_Score[1]+=Demand_Store[i][1];
   }
   if (Demand_Store[i][1]<0)
   {
      Pattern_Score[2]+=Demand_Store[i][1];
   }
}
```

Figure 4.11: Code for calculation of basic objective function.

4.5.3 Search Steps

The optimizing search now begins through redistribution of the initially assigned staff amongst the shifts defined using the input staff roster. This is achieved by selecting every possible pairing of available shifts and, if possible, attempting to move a single staff member from the first of the pair to the second; in this way the move used is of the first type listed in Section 4.5.2. After each of these trial moves is made the roster score is recalculated using the 'Score' routine to take account of the change that has been made, this score is then stored within a unique element of an array and the move is reversed. In this way then every possible roster that may be reached through a single move from the initial one is encountered and has its objective function score recorded. Thus in this way the entire local neighbourhood surrounding the current solution is investigated in the same way as would be understood for a neighbourhood search algorithm. The best of these moves is then selected to be carried out as a more permanent adjustment to the roster; in this case a best move is taken to be that resulting in the greatest improvement to the current roster score towards optimality.

For an example of how this would work consider a scenario in which there are 6 shifts available for staff redistribution. Each possible coupling of these 6 shifts would then be equivalent to a potential move of a staff member from one of the shifts to another. Note that the pairing (x, y) is distinct from the pairing (y, x) here as staff will be moved from the first selected shift to the second; so from x to y in the case of (x, y) and from y to x for (y, x). Thus for this example, there are 36 possible couplings including 6 essentially null couplings in which staff members are moved both from and to the same shift; these 6 would be represented using pairings of the form (x, x). A grid of the style displayed in Figure 4.12 illustrates a structure capable of recording information regarding each of the possible pairings in this scenario.

	1	2	3	4	5	6
1	■					
2		■				
3			■			
4				■		
5					■	
6						■

Figure 4.12: A grid used to record the effects of the moves associated with shift couplings conducted as part of the redistribution stage in the tabu search optimization method.

Each of the moves associated with these couplings would then be conducted and the quality of resulting rosters would be assessed according to the objective function described in subsections 4.4.5 and 4.5.2. Consider the example described in Figure 4.13 in which an initial staffing roster is described alongside a grid containing roster scores that result from moves using the associated couplings. In this example some couplings result in trying to move staff members from shifts where there are none available; for example any move that tries to take a staff member from shift number 3. This is a physical impossibility and as such the search algorithm assigns a very large score to the coupling to negate the possibility of use; note that for this example higher quality rosters would be deemed as those with a lower score. Then the most improving move would be given by the coupling (1, 3) representative of the movement of a single member of staff from shift number 1 to shift number 3. This move is attributed the lowest score when assessed using the objective function and as such represents the move resulting in the greatest reduction in unmet demand.

Shift	Staff
1	5
2	3
3	0
4	2
5	4
6	1

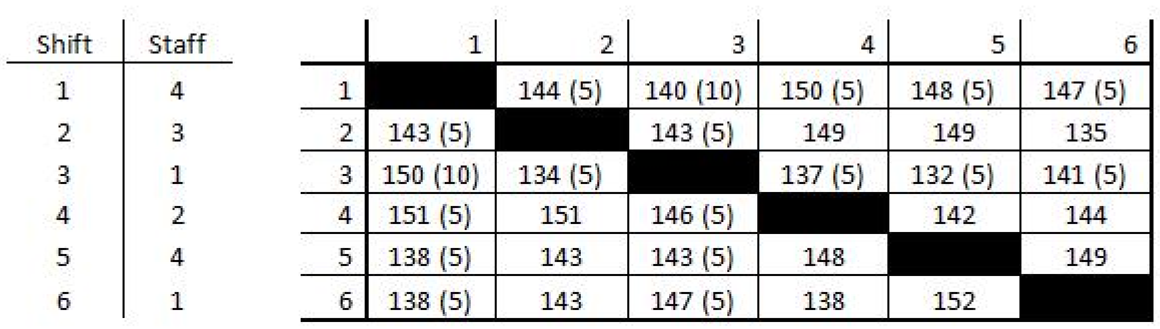
	1	2	3	4	5	6
1	■	148	143	152	150	145
2	150	■	146	150	152	151
3	1000	1000	■	1000	1000	1000
4	149	148	151	■	147	144
5	151	150	150	153	■	147
6	152	157	155	153	154	■

Figure 4.13: Example initial staffing roster and score memory structure resulting from staff redistribution based on all possible shift couplings.

This process continues until no further improving move can be found, with a few additional stages that are crucial to the success of the search. Firstly upon conducting a permanent move, both of the involved shifts are assigned a tabu tenure which is set to some pre-specified value. When future moves are considered this tenure is added to the score of any potential rosters that will result from the use of either of these particular shifts. This will result in a search that encourages diversity through forcing the selection of a wider variety of shifts as opposed to making repeated use of the same ones. However, applying the tenure as a penalty (instead of using it to simply rule out potential moves) will ensure that particularly favourable moves which prove to be sufficiently improving as to overcome the tenure based penalty may still be considered. For the purposes of the searches conducted in this research it was desired to encourage localized intensification from the effects of tabu tenure, accordingly then it was necessary to use a smaller value for the penalty; larger values would encourage a broader but less detailed search. Through experimentation it was found that a value of 5 was suitable for this.

As more iterations of the search occur any further shifts used for permanent moves will also be assigned the same tabu tenure. Additionally the tenure value assigned to each of the previously tabu shifts will be decreased in order to begin allowing them to be more readily used again; this also will result in the reduction of the associated penalty for their use. Such decrease is achieved through reduction of tenure penalties by 1 at each subsequent iteration of the search; for example, if a move were to be associated with a penalty value of 5 then at the next search iteration this penalty will decrease to 4. This system should result in a search that selects a wide variety of moves and so covers a larger area of the solution space than a simple steepest descent algorithm would.

So continuing the previous staff redistribution example, now taking into account a tabu tenure penalty function, a result similar to that shown in Figure 4.14 could be achieved. All roster scores that involve the use of either shift number 1 or shift number 3 are marked with a score penalty; these penalty values are indicated in parentheses adjacent to the unadjusted score. There are several ways in which the tabu tenure could be applied at this stage. In the example provided a penalty value of 5 has been attributed to any score resulting from a coupling involving either of the shifts numbered 1 or 3; when shift 1 and 3 are coupled together this is cumulated to a penalty score of 10. The benefit of this approach is that it will assign a high penalty to both of the couplings (1, 3) and (3, 1), these representing a repetition of the same move and the reverse of the previously conducted move respectively. With the example of Figure 4.14, the most improving move available within it would be the adjustment caused by coupling (2, 6) which gives a score of 135; this coupling represents the movement of a staff member from shift number 2 to shift number 6. However if tabu tenure penalties had not been applied then the most improving move would have been given by the coupling (3, 2) with a score of 134; with the penalty from the previously conducted move included this score is boosted to 139 making (2, 6) a more attractive coupling.



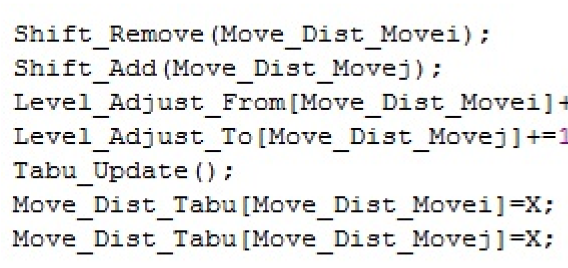
Shift	Staff
1	4
2	3
3	1
4	2
5	4
6	1

	1	2	3	4	5	6
1	■	144 (5)	140 (10)	150 (5)	148 (5)	147 (5)
2	143 (5)	■	143 (5)	149	149	135
3	150 (10)	134 (5)	■	137 (5)	132 (5)	141 (5)
4	151 (5)	151	146 (5)	■	142	144
5	138 (5)	143	143 (5)	148	■	149
6	138 (5)	143	147 (5)	138	152	■

Figure 4.14: Example staffing profile and scoring memory structure after an iteration of the staff redistribution stage including sample tabu tenure penalties.

Alternatively tenure penalties could be applied only to moves that match one or both of the conducted coupling elements more closely by penalizing moves that would be those going from shift 1 and those going to shift 3 only. Thus moves of the form (1, x) or (x, 3) would be the only ones penalized, where x is any other shift number; this would mean that the reverse move (3, 1) would have no penalty attributed to it whatsoever. Indeed it is possible to be equally selective only limiting moves that would encourage the reverse of that conducted to occur, thus penalizing such that only moves of the form (x, 1) and (3, x) suffer penalty. This is beneficial if it is likely that repeated performance of the same move would provide a high benefit to final solution quality; with potentially a heavier and heavier penalty being attributed to the reverse move, strongly preventing its occurrence. Indeed if desired then penalties could be applied to only very specific moves based on the previously used coupling. For example by penalizing both couplings that use the same 2 elements as were previously selected with a tenure penalty; in this case this would be the couplings (1, 3) and (3, 1), the coupling associated with the same move and that associated with its reverse. A final consideration is to whether tabu tenure penalties would be applied cumulatively or in replacement of each other where necessary. This would mean that in a situation where a tabu-active element is allowed to be used, most likely through some aspiration criteria, that the tabu tenure currently applied to that element would either be reset to the standard value or increased by the base tabu tenure amount.

Within the C++ based search used as part of this research a series of 3 different moves are considered, requiring different methods of application of tabu tenure penalty. The first type of move, redistribution of a staff member from one shift to another, is penalized through the application of 2 different tenure penalties. A first penalty is applied to the element representing the shift from which a staff member is moved and a second penalty is applied to the shift to which the staff member is then moved. In this way individual donators or receivers of staff are penalized within this stage of the search process. This is preferable to simply penalizing all involved elements as there is potential for a strong donator to later become a strong receiver of staff (or indeed a receiver could equally become a donator) and with a penalization for such an element whilst it was active in its previous role, potentially improving moves could remain unconsidered. For the second and third move types that are involved with increase or decrease in shift starting and ending time penalties are applied to whichever shift is selected for starting time adjustment. An example of the code used in order to implement this penalization is illustrated in Figure 4.15. This segment of code is implemented after all trial moves have been conducted and only if an improving move is found. The first 2 lines implement the staff removal and addition processes that carry out the best found staff member re-assignation process, following this records that these moves have been conducted are logged with the increase of array values recording the number of times each search element has been used. Finally the tabu penalties previously implemented are updated through a 'Tabu_Update()' subprocess (which reduces all prior tabu penalties by 1) and new penalties are applied to the elements selected for the current move by updating of 'Move_Dist_Tabu[i]' values where the i^{th} element of this array records the penalty attributed to element i.



```
Shift_Remove(Move_Dist_Movei);
Shift_Add(Move_Dist_Movej);
Level_Adjust_From[Move_Dist_Movei]+=1;
Level_Adjust_To[Move_Dist_Movej]+=1;
Tabu_Update();
Move_Dist_Tabu[Move_Dist_Movei]=X;
Move_Dist_Tabu[Move_Dist_Movej]=X;
```

Figure 4.15: C++ code used to apply tabu tenure penalty in staff redistribution search.

When an optimum solution is found it is not possible to tell immediately whether it is a local or a global one. In order to compensate for this it is necessary to force the search to attempt to find other solutions by some method in which the current roster is perturbed away from its solution and the search continued. In order to achieve this, a continuous log of the permanently conducted moves is maintained and upon reaching an optimum solution the previous few moves are undone; essentially this returns the search process to a previous state. This logging method is as shown in Figure 4.15. Prevention of the process leading straight back to the same solution as previously is achieved through retention of assigned tabu tenure values. Thus those moves that lead directly to a return to the same optimizing solution, being the most recently conducted, will have high tabu tenure values and still suffer from large penalties. This will discourage the repeat selection of the same sequence of moves and will help prevent the search becoming trapped at a single locally optimizing result. After a sufficient number of optima have been found that offer no improvement over previous solutions the best is declared to be a strong optimum and the search procedure continues onto the next phase. Note that at this stage the best result that can be found through adjustment of staffing levels around the base shift pattern has been found; the next stage will result in adjustment to the shift structure itself.

This next stage is to consider the adjustment of shift starting times which is achieved through a similar process as that for staff redistribution. The increase of every shift start time by a single hour is considered individually with the resulting rosters each having their score calculated as in the previous stage. Again the most improving of these moves is then conducted as a permanent change to the roster and a tabu tenure value is assigned to the shift which has been moved for the same reasons as before. Indeed the same process is observed as regards application of tabu tenure as a penalty to the score of potential future moves; further the same approach is taken for perturbation and continued search upon the discovery of an optimal solution. The very similar third stage following this is to perform the exact same procedure with the only change being that potential moves result from a reduction in the start time of shifts by 1 hour instead of an increase by the same amount.

For a move involving the increase or decrease of a shift start time, only a single element will be adjusted. This will mean that only one element will obtain tabu-active status and be assigned a tabu penalty per iteration of the search process. Thus tabu tenures for these 2 search stages are stored in a simple array, named $Move\_Times\_Tabu[]$; note that only 1 such array is necessary as only one search type will occur at a time, with all values in the array being zeroed between searches allowing it to be reused.

For a final stage, the optimizing search first conducted, in which staff numbers are redistributed amongst the defined shifts, is performed once more to take maximum advantage of any start time adjustments that may have occurred. At this point the program will, if set to do so, generate one or more of the output files described within subsection 4.4.4 and then terminate.

4.6 Results

4.6.1 Tabu Tenure Variation

For each problem considered the search procedure is run a number of times using variation in the tabu tenure penalty values attributed to each move as it is performed. The intent of this was to attempt to investigate any relation between the tenure values and the quality of solution produced and the number of search iterations required to attain it. The tabu search methodology was applied through use of the 3 defined move types in the order of staff redistribution followed by shift starting time increase and finishing with shift starting time decrease. Figure 4.9 provides a sample of such output using variation in tabu tenure penalties for the NL region of Leicestershire considered over a full calendar week. The quality of roster scores contained within this table represent the average number of incidents that would be expected not to be immediately met over the week considered; this is quantified as all unmet demand using the input demand profile and the final output staffing roster. In this table 'Tenure 1' and 'Tenure 2' refer to the numeric penalty values that are attributed to the optimizing moves conducted in order to temporarily inhibit their repeated use. Thus in Table 4.9 the region with tenure 1 varying from 3 to 6 and tenure 2 from 11 to 13 yields the highest quality solution purely in terms of minimization of unmet demand. The secondary shaded region indicates those solutions that are of a higher quality than that achieved through simply performing a steepest descent type search; this being equivalent to the tabu search but with all tenure values being set to 0.

Further investigation of tenure variation was conducted by performing a series of tests upon each of the 18 trial problems. Tabu tenures were allowed to take every integer value from 0 to 20 for each stage of the search with the best resulting score discovered from this series of searches recorded and presented here alongside the steepest descent results in Table 4.10. In cases where the wide tabu search offers no improvement over steepest descent it was commonly seen that a large range of potential tabu tenure values all resulted in providing the same final optimum. This is strongly suggestive that in these cases there are only a few moves that may be performed before a strongly attractive optimizing result is found; indeed the effects of the tabu tenure may never come into use in these problems. In contrast, where the wide tabu search outperforms steepest descent it is often seen that a very small range of values provide the strongest optimum solution; however a larger range of tenure values are still seen to outperform the steepest descent approach.

Table 4.9: Table collecting best roster scores through variation in tabu tenure penalties when applying the tabu search method to the NL region trial problem.

		Tenure 1							
		3	4	5	6	7	8	9	10
Tenure 2	0	34.2477	34.3023	34.3023	34.3023	34.3023	34.3023	34.3023	34.3023
	1	34.2477	34.3023	34.3023	34.3023	34.3023	34.3023	34.3023	34.3023
	2	34.2477	34.3023	34.3023	34.3023	34.3023	34.3023	34.3023	34.3023
	3	34.2477	34.3023	34.3023	34.3023	34.3023	34.3023	34.3023	34.3023
	4	34.2477	34.3023	34.3023	34.3023	34.3023	34.3023	34.3023	34.3023
	5	34.2477	34.3023	34.3023	34.3023	34.3023	34.3023	34.3023	34.3023
	6	34.2477	34.3023	34.3023	34.3023	34.3023	34.3023	34.3023	34.3023
	7	34.2477	34.3023	34.3023	34.3023	34.3023	34.3023	34.3023	34.3023
	8	34.2477	34.3023	34.3023	34.3023	34.3023	34.3023	34.3023	34.3023
	9	33.8397	33.8943	33.8943	33.8943	33.8943	33.8943	33.8943	33.8943
	10	32.5209	32.5209	32.5755	32.5755	32.5755	32.9973	32.9973	32.9973
	11	30.1584	30.1584	30.1584	30.1584	30.8653	30.8653	30.8653	31.765
	12	30.1584	30.1584	30.1584	30.1584	30.8653	30.8653	30.8653	31.765
	13	30.1584	30.1584	30.1584	30.1584	30.8653	30.8653	30.8653	31.765
	14	32.0902	32.0902	32.0902	32.0902	32.0902	32.512	32.512	32.512
	15	32.3021	32.3567	32.3567	32.5945	32.5945	33.0163	33.0163	33.0163
	16	34.9447	34.747	34.747	34.747	34.747	34.747	34.8016	36.1408
	17	34.9447	34.747	34.747	34.747	34.747	34.747	34.8016	36.1408
	18	34.9447	34.747	34.747	34.747	34.747	34.747	34.8016	36.1408
	19	34.9447	34.747	34.747	34.747	34.747	34.747	34.8016	36.1408
	20	34.9447	34.747	34.747	34.747	34.747	34.747	34.8016	36.1408

An alternative explanation for this behaviour could be that many (or even all) moves that occur within the search are very strong in leading towards optimization. This would mean that the tabu tenure penalties could be too low to have any real impact upon the path taken to optimization; the optimizing moves in this case being regularly so strong as to overcome any penalty placed upon them. It could also be the case that a specific sequence of moves lead to optimization that would fall in such a way that tabu tenure will never result in the disqualification of an otherwise optimizing move; this particular scenario is highly improbable however.

With 12 of these 18 results showing an improvement using tabu search over steepest descent (with both searches providing the same quality solution in the remaining 6 cases) it is quite strongly suggested then that careful selection of appropriate tabu tenure penalty values can have great effect on the strength of tabu search. Indeed 16 of the searches see an improvement from the simple tabu search results shown in Table 4.13 with the other 2 showing no change to the best solution. This is likely due to the fact that in these cases there is very little adjustment available; thus making it also probable that in finding the same result, each of

Table 4.10: Roster score comparison for each of the Leicestershire policing regions using steepest descent method and wide tabu search.

Region	Steep. Desc.	Wide Tabu Search	Region	Steep. Desc.	Wide Tabu Search
City	41.0713	39.2445	*NH*	18.2212	18.2212
North	31.7892	28.2949	*NL*	33.4453	28.6547
South	31.0179	26.7015	*NM*	2.39284	2.39284
CB	25.986	25.986	*NR*	5.6325	5.6325
CH	54.7174	53.8537	*NW*	61.9436	60.1158
CK	6.72969	6.72969	*SB*	27.9267	26.1
CM	23.1931	21.6322	*SH*	68.7975	65.1906
CN	3.42615	3.39677	*SM*	5.58638	3.38512
CW	13.8965	13.0519	*SW*	5.13775	5.13775

the 3 searches have found a global optimum. Figure 4.16 displays a graphical representation of the best rosters found using the steepest descent approach and tabu search methodology. In this graph the shaded region is indicative of expected demand, the solid line illustrates the result from tabu search and the dashed line the result from steepest descent. Though there is not a strong difference in these rosters (as evidenced by the similarity in objective function value) it is shown that they differ primarily in the times at which shift overlaps occur. This is suggestive that the tabu search is better able to monopolize upon the potential benefit that such periods within a roster may have upon solution quality. From these results it can be concluded that tabu search offers a viable avenue for exploration of the solution space available for problems of this nature; though importantly it should be noted that it is far more well suited to problems that have larger potential solution spaces. For the benefit of comparative research into the effectiveness of addition of or adjustment to search mechanisms used, it is suitable to fix the tabu tenure penalty to some constant value so long as the same value is used for each method being compared. In this way, future trials were conducted using a fixed tabu tenure penalty of 5 for all problems allowing immediate comparison between the increased effectiveness provided by techniques considered.

4.6.2 Full Search with Staff Redistribution

It was important to validate the notion that re-performing the staff redistribution search technique after varying the shift start times did indeed provide a benefit to the quality of solutions. In order to do so, all 18 of the test problems were conducted using the full search procedure described in subsections 4.3 and 4.5 and then re-conducted following the same procedure minus the second staff redistribution stage. firstly the search distributes staff as

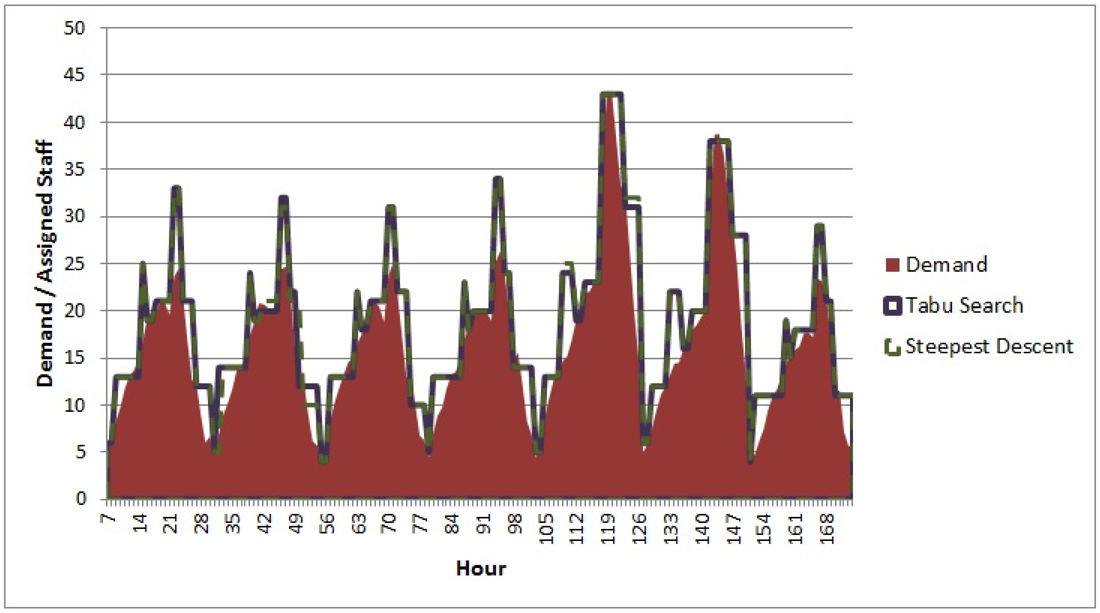


Figure 4.16: Graph comparing best rosters obtained through steepest descent and tabu search methods.

best it can across initially defined shifts using the first type of move and then proceeds to attempt increase and then decrease of shift starting times with the second and third move types respectively; each of these move types is used until a point is reached at which no further improvement may be found, whereupon the search proceeds to the next move type. With the additional staff redistribution stage, the search will continue until all 3 of the move types have been used fully and will then attempt to use moves of the first type again in redistributing staff across the redefined shifts. For each stage of the search process in every one of the test problems the tabu tenure penalty was set to a constant value of 5. Further the search was set to terminate upon discovery of a first optimizing result and did not use the multiple minima search diversification as previously discussed. The results of this are summarized within Table 4.11, where the best found locally optimizing solutions for the searches running both with and without the additional staff redistribution stage are presented for each of the 18 trial problems. It is quite clear from this that in many cases (11 out of the total 18) an improvement to the final optimum solution is found; this improvement ranging anywhere between 0.22% and 10.97%, with an average of 2.34% taking all 18 comparisons into account. This would strongly suggest that, as would be expected intuitively, the inclusion of the second staff redistribution step is beneficial to the quality of final solutions produced by the search method.

To aid in illustration of these differences, Table 4.12 contains example weekly rosters for the

Table 4.11: Results from testing tabu search both with and without a secondary staff redistribution search stage.

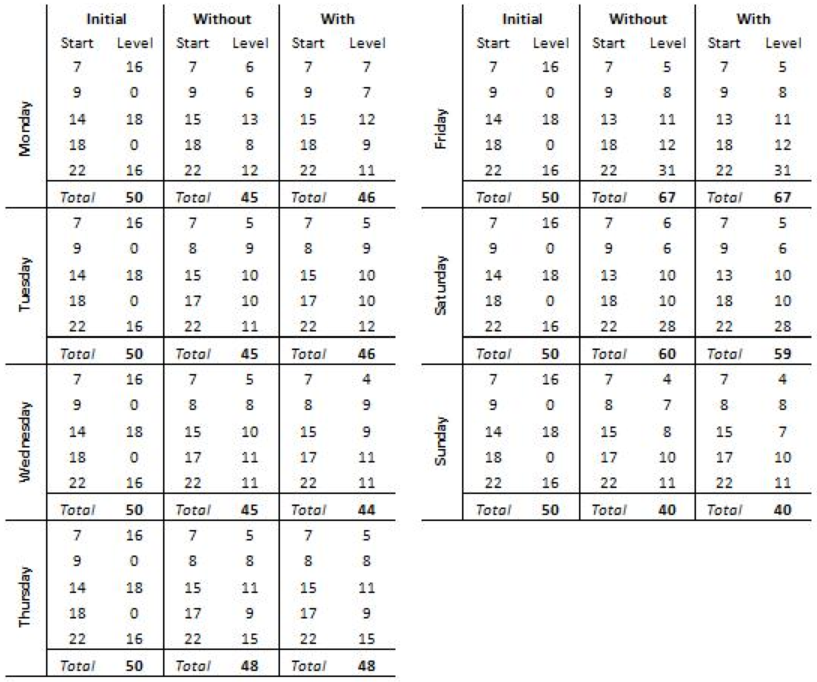
Region	Without	With	Region	Without	With
City	42.8179	39.784	NH	20.3039	20.3039
North	34.1045	31.7311	NL	34.3023	32.7987
South	34.0722	30.3353	NM	2.48192	2.39284
CB	26.8237	26.3009	NR	6.11011	6.11011
CH	55.8845	55.8845	NW	62.083	61.9436
CK	6.79866	6.72969	SB	28.8637	27.9267
CM	25.3516	25.3516	SH	69.243	68.7975
CN	3.6575	3.6575	SM	4.7377	4.7377
CW	15.2636	14.954	SW	5.23825	5.23825

City region. These rosters describe results gained through application of the tabu search procedure either with or without the additional redistribution stage in addition to the initial profile used to set up the problem. Important to note is that the total number of staff assigned over the whole week remains the same at 350 staff members. Currently no potential to reduce staffing numbers is included as part of the objective of the search and accordingly the problems are designed to instead search for best use of available staff in a situation where not all demand may be immediately met. Though the total staffing levels are the same Table 4.11 shows that in the majority of cases the results produced with the additional staff redistribution stage are of better quality than those produced without it. Indeed the City problem is shown to see an improvement from 42.8179 to 39.784 surplus demand remaining; the positioning of staff members as illustrated in the 'without' and 'with' columns of Table 4.12 are those representative of the rosters resulting in these objective function scores respectively. The 'without' columns will dictate the state of the search process that is reached before the additional staff redistribution stage is occurred; thus differences between the 'without' and 'with' columns will illustrate all optimizing moves conducted within this final stage.

4.6.3 Comparison with Steepest Descent

It is possible to compare the tabu search application considered with results that would have been attained using the same search method but with a simplistic 'steepest descent' approach. This method would be identical to performing the same search with no tabu tenure restrictions in place whatsoever upon the search; this would mean that the best move would be selected at each stage until the first optimum solution is found, at which point

Table 4.12: Comparison of initial roster with rosters resulting from application of the optimizing search method with and without the additional staff redistribution stage on the City trial problem.



	Initial		Without		With	
	Start	Level	Start	Level	Start	Level
Monday	7	16	7	6	7	7
	9	0	9	6	9	7
	14	18	15	13	15	12
	18	0	18	8	18	9
	22	16	22	12	22	11
	Total	**50**	*Total*	**45**	*Total*	**46**
Tuesday	7	16	7	5	7	5
	9	0	8	9	8	9
	14	18	15	10	15	10
	18	0	17	10	17	10
	22	16	22	11	22	12
	Total	**50**	*Total*	**45**	*Total*	**46**
Wednesday	7	16	7	5	7	4
	9	0	8	8	8	9
	14	18	15	10	15	9
	18	0	17	11	17	11
	22	16	22	11	22	11
	Total	**50**	*Total*	**45**	*Total*	**44**
Thursday	7	16	7	5	7	5
	9	0	8	8	8	8
	14	18	15	11	15	11
	18	0	17	9	17	9
	22	16	22	15	22	15
	Total	**50**	*Total*	**48**	*Total*	**48**
Friday	7	16	7	5	7	5
	9	0	9	8	9	8
	14	18	13	11	13	11
	18	0	18	12	18	12
	22	16	22	31	22	31
	Total	**50**	*Total*	**67**	*Total*	**67**
Saturday	7	16	7	6	7	5
	9	0	9	6	9	6
	14	18	13	10	13	10
	18	0	18	10	18	10
	22	16	22	28	22	28
	Total	**50**	*Total*	**60**	*Total*	**59**
Sunday	7	16	7	4	7	4
	9	0	8	7	8	8
	14	18	15	8	15	7
	18	0	17	10	17	10
	22	16	22	11	22	11
	Total	**50**	*Total*	**40**	*Total*	**40**

the search terminates. The steepest descent will follow the exact same move sequence as the tabu search, in which it will proceed through the 3 types of move in sequence until no further improvement to solution quality is found. Scores for rosters found in such a 'steepest descent' method are contained within Table 4.13 alongside the scores for the tabu search method; further note that both the steepest descent and tabu search make use of the additional staff redistribution stage in these results. For the tabu search method all tabu penalty values assigned were again set to the constant value 5. Though in some of the trial problems this was seen not to be the strongest performing tenure penalty, it was a suitably high performance value that when fixed to a constant allowed for simpler batch runs of trials to be performed.

These results suggest that the tabu search is weaker than simple steepest descent search when used upon this series of problems in this fashion, providing improvement in only 5 of the 18 problems. Indeed in 8 of the 18 tests the steepest descent algorithm is seen to outperform the tabu search. Importantly however it is in cases where there are greater staff levels available for distribution amongst defined shifts that the tabu search is seen to be the strongest performer. This is suggestive that the tabu search is only useful in problems in which a greater level of system variation is possible. This is almost an intuitive result

Table 4.13: Roster score comparison for each of the Leicestershire policing regions using steepest descent method and the simplest tabu search approach.

Region	Steepest Descent	Simple Tabu	Region	Steepest Descent	Simple Tabu
City	41.0713	39.784	*NH*	18.2212	20.3039
North	31.7892	31.7311	*NL*	33.4453	32.7987
South	31.0179	30.3353	*NM*	2.39284	2.39284
CB	25.986	26.3009	*NR*	5.6325	6.11011
CH	54.7174	55.8845	*NW*	61.9436	61.9436
CK	6.72969	6.72969	*SB*	27.9267	27.9267
CM	23.1931	25.3516	*SH*	68.7975	68.7975
CN	3.42615	3.6575	*SM*	5.58638	4.7377
CW	13.8965	14.954	*SW*	5.13775	5.23825

as tabu search routinely restricts the amount of available moves at each search iteration; in situations where there are fewer moves then the restriction imposed may be too severe. Indeed, in Figure 4.16 wherein a graphical comparison of rosters resulting from tabu search and steepest descent application are presented, it is seen that the tabu search is able to improve solution quality through optimization of the regions at which shift overlaps occur. In regions in which there is less staff flexibility it is likely that adjustment to these overlaps will provide lower quality solutions, reducing the potential impact that tabu search may provide. It may then be possible to counter this and encourage the wider application of tabu search through variation of the tabu tenure penalty values; if moves are freed sooner then it may be more applicable to these problems with fewer available moves. This would also suggest that an increase in tabu tenure values could potentially force a more diverse search and thus greater success rate in searches with a very large amount of available moves; the top 5 such problems being the ones showing improvement using the simple fixed tenure tabu search.

4.7 Summary

The work conducted in this chapter has resulted in the creation of a tabu search method that is suited to the Police staff rostering problem. This tabu search approach has the following key factors:

- Follows a pattern of move application in which staff are first distributed across defined shifts, the starting times of these shifts are then adjusted and a secondary staff redistribution search is implemented. In the terms of the moves as defined in Section 4.4.2 this is equivalent to performing moves of type 1, type 2, type 3 and finally type 1 again.

- Use of tabu tenure penalties based upon the recency with which individual search elements were last used to conduct a move within the search process. The most commonly seen most well suited tenure penalty was 5, this meaning that used elements will remain penalized for the following 5 search iterations.
- Performs until a solution is found such that it has no neighbouring solution of better quality.

4.8 Potential Improvements

A potential route for development comes in the form of a different type of objective function. Leicestershire Police make use of specific target figures for their attendance to incidents of grades 1 and 2; these targets being the attendance of 85% of grade 1 incidents within 15 minutes and attendance at 80% of grade 2 incidents within 60 minutes. In defining a series of incidents that occur throughout a calendar day it should be possible, through consideration of available Police staff, to provide some estimate of the percentage success rate in meeting these targets. Further, using this as an objective function would create a more directly applicable and understandable measure of success to the Police business.

One of the areas which provides most scope for improvement is that of inclusion of diversification techniques; these should encourage a much broader search of the potential solution space and begin to take full advantage of the strengths of tabu search. It could also prove beneficial to begin to take into consideration intensification stages that search more thoroughly highly promising looking areas within the solution space. With inclusion of either or both of these additions it would be expected that solution quality could vastly improve in many of the trial problems used here. Chapter 5 discusses options for both of these categories of search augmentation and also describes other adjustments made to the tabu search along with results from their application.

4.9 Conclusions

This chapter has shown that tabu search is a valid option for use in optimization of the target staffing to demand problem presented by Leicestershire Police. Major topics covered in this chapter are:

- Introduction and discussion of key aspects of the tabu search algorithm.

- Creation of a tabu search algorithm that is tailored to the problem of resourcing to demand.
- Presentation of C++ code used in implementation of the customized algorithm.
- Testing of search parameters and comparison against other search techniques.

Through this testing and the results presented within Section 4.6 it was seen that:

- The tabu search method created is capable of outperforming a steepest descent based local neighbourhood search algorithm.
- For each of the trial problems considered a particular range of tabu tenure penalty values exist that will result in provision of best solutions in each case.
- The ordering of moves trialled by tabu search was an important factor and it was found that an additional staff redistribution stage within the search resulted in improved solution quality.

There is still potential to further enhance the quality of the search algorithm through implementation of more advanced techniques available to the tabu search; for example diversification and intensification techniques. These techniques should be able to address the strict limitation currently placed upon the search through the rigidly defined shift structure over which staff are optimized and indeed should allow the discovery of a best series of shifts with a best assigned staffing allocation. Work detailing this is included within the following chapter.

Chapter 5. Tabu Search Advanced Implementation

5.1 Introduction

The tabu search method is able to reach beyond the scope of a basic local neighbourhood search or steepest descent search through the inclusion of search process modifiers. These modifiers act based upon knowledge gained from previous search iterations or even from an entire previously conducted search. The method in which this information is utilized and precisely how it is applied in order to perturb a tabu search (either in progress, or as it initiates) will vary with application. All such methods will however fall into the category of either a diversification technique or an intensification technique. In brief, the use of a diversification method will encourage perturbation of the search procedure in such a way that a wider region of the potential solution space is investigated. The alternative, intensification approach, instead aims to identify particularly strong regions of the solution space and more thoroughly investigate them by constraining the search in some manner. This chapter will discuss both of these approaches to tabu search modification in the pursuit of improving the tabu methodology for Police application in more detail and will describe applications of both these methods to the tabu search algorithm defined in Chapter 4. Results of the application will be provided and it will be shown that diversification at least offers potential for improvement to search quality.

5.2 Diversification

5.2.1 Introduction to Diversification Techniques

The primary objective achieved through successful application of a diversification technique is to increase the region of the potential solution space considered whilst performing a tabu search. Through consideration of a more diverse population of solutions the search should be discouraged from becoming trapped at any particular strong local attractor. To achieve such diversification, the search must be either forced or encouraged to investigate different areas within the solution space than it would usually encounter. This may be accomplished through perturbation to the search method that may occur either as an active component throughout or as an additional phase within the search initialization process. Within this research, 2 particular approaches have been considered, a newly constructed ‘multiple minima diversification’ and a more traditional ‘restart diversification’.

5.2.2 Multiple Minima Diversification

A fairly commonly used stopping criteria within optimizing search is the condition that a specific number of iterations has passed with no improvement provided to solution quality. This number of iterations should be, by design, sufficiently large so as to be confident that a wide enough search of the solution space has been performed in providing a strong optimizing result. However tabu search will not be able to perform adequately with such an option as the basic search acts as a modified local neighbourhood search in finding a single attractor. An alternative method used in this research is to instead perturb the search process at any point at which a locally optimizing result is found; resulting in what is named a 'multiple minima search'.

An example of perturbation that occurs throughout the search process that was considered and tested within this research is a multiple minima search. The base idea behind this diversification is to perturb the search away from any discovered locally optimizing solution and encourage it to find another alternative solution within some local vicinity; accordingly this perturbation is set to occur upon discovery of a locally optimizing result. Perturbation was achieved through reversal of a number of the most recent iterative steps taken towards optimization; effectively this results in returning the search to a previous state that occurred a set number of iterations ago. It is important to prevent the search from proceeding along the same route to optimization, which would result in rediscovery of the previous optimizing solution. In order to achieve this, the tabu tenures associated with the moves subject to reversal in the perturbation are retained. Thus the search will return to a prior state and with the previous optimization path being heavily penalized it will be encouraged to explore an alternate route ideally ending in a different solution. Each solution discovered is logged before perturbation with the relative strengths of each compared in order to provide the best solution upon final search termination; the strength of each solution being quantified through use of the surplus demand based objective function.

The results from application of this method will provide a benchmark figure from which other more complex techniques may be assessed. First the search algorithm will proceed until no more improving moves can be found in the neighbourhood surrounding the current best solution. At this point the current best solution will be perturbed and the search will then continue. In order to perturb the best solution the 5 most recently occurring moves will be reversed forcing the search back along the search path to a previously considered solution. However whilst the most recent moves are essentially undone their tabu tenures will be retained; this being the reason for reversal of only the 5 most recent moves, as with a maximum tenure of 5 it is guaranteed that each of the reversed moves will still have some penalty associated with its repeated use. This will discourage the search from making the same choices of move and providing the same best solution again due to the heavy score penalties that would be involved in reusing the still tabu-active elements. This process of obtaining an optimizing result, back-tracking from it and then continuing the search is repeated 9 times so that in total 10 locally optimizing solutions are found; the best of these would then be selected as the most optimal result. A series of searches were conducted allowing up to 1000 minima to be discovered by this process and in no cases was an improving solution discovered beyond the 10^{th} generated; from this it was chosen to terminate the multiple minima search after 10 optima are found in order to reduce calculation time.

5.2.3 Restart Diversification

A restart diversification approach is implemented through perturbation of the starting conditions of a search by considering the results of a previously conducted search. Such perturbation is achieved through selection of a few of the system variable parameters and forcing the solution into a new region of the solution space by setting them to infrequently used values. These variable parameters would be any part of the solution that may be adjusted using any of the search transformations (or moves) defined for use by the search procedure. Within this research, a potential solution is a staffing roster that covers a desired period of time through definition of a series of shift starting times, shift durations and staffing allocations. Of these, the staffing allocations are highly variable within the tabu search itself and would not be well suited to restart diversification as any changes caused by it would likely be undone early within the iterative search process. Though shift duration could be adjusted, it is unlikely in the real world to stray far from a preset value and indeed there are many potential conflicts with the working time directive that may arise from its adjustment. Thus the only suitable option is to allow changes to be made to shift starting times by restart diversification.

The next challenge in order to apply the method was in identification of the elements that will be perturbed upon restarting the tabu search. As it is sought to investigate previously unexplored regions of the solution space, it would be reasonable to attempt to identify (and thus select for perturbation) those shifts that are least used within a previous search. Within the 2 search stages that are involved with trial adjustment to shift starting times, a count was recorded for each shift that detailed the amount of times it was selected for either an increase or decrease in starting time. Those with the lowest count would be those that were least selected (or used) throughout the search. It is reasonable to assume from this that a perturbation to these least used shift starting times would force the search to enter a previously unexplored region of the solution space. Indeed, if only those shifts that were completely unused within the previous search were to be subjected to restart diversification then it would be guaranteed that unexplored solutions would be considered during the subsequent search.

With a series of shifts defined that are suitable for restart diversification it was necessary to define what perturbation would be applied by the method to each of them. For this, a range of values was defined which could be applied as a direct addition to shift starting times; negative values were also included in the range to allow for starting time reductions. For each shift selected for perturbation a value would be randomly generated from within this range and applied to the starting time. For example, with a range of $\{-1, 0, 1\}$ a selected shift could either have its starting time decreased by 1 hour, left unchanged or increased by 1 hour. It could potentially be of benefit to consider a weighting to the probabilities used in selection of perturbation values however it was seen within this work that a uniform distribution was well suited; thus each value in the range having equal probability of selection. With such a distribution the chance of a shift remaining unperturbed is lower than that of it being perturbed and importantly the chance of perturbation increasing shift starting time is equal to that of perturbation decreasing starting time. Through this then, variation will commonly be encouraged within selected shifts and indeed with equal possibility of increase or decrease of starting time it would be expected that overall system structure variations would remain within some neighbourhood of the initial solution. Thus by design of the initial solution, some expected level of control will exist within final generated system optima.

5.3 Intensification

5.3.1 Introduction to Intensification Techniques

The most commonly used form of tabu search intensification technique is to discover potentially profitable regions of the solution space and to focus the search on these regions as it progresses. This is usually achieved through the continuous updating of a list of the most recently conducted moves, with this list being used in order to indicate repeat use of any given search element. Upon identification of such a search element, the search will progress using a restriction on the available moves considering only those that involve the identified element until a locally optimizing result is found. When an optimizing solution is found using this intensification, the restriction upon available moves is lifted and the search will continue as usual.

A method of this style was not appropriate within this research as it was seen to work counter to the multiple minima diversification approach. Indeed use of this basic intensification was seen to harm the quality of solutions found in this case by essentially forcing the search to favour strong local optima. Indeed trial searches conducted upon the 18 LPU and BCU based problems using such an intensification technique were seen to be often outperformed even by the steepest descent approach mentioned in Chapter 4. Results from such a series of trials are displayed within Table 5.1 where it is clearly seen that the tabu search with intensification is outperformed in 12 of 18 cases and indeed only matches the result of steepest descent in the other 6. Entries in the 'Tabu' columns indicate roster scores obtained through use of standard intensification and those in the 'Steepest Descent' columns show scores from application of the steepest descent technique. Thus in order to investigate the potential benefit of search intensification it was necessary to consider alternative forms of the technique and indeed to create an entirely new approach, namely restart intensification.

Table 5.1: Roster score comparison for each of the Leicestershire policing regions using steepest descent method and tabu search with intensification.

Region	Steepest Descent	Tabu	Region	Steepest Descent	Tabu
City	41.0713	41.5603	*NH*	18.2212	19.8124
North	31.7892	32.2841	*NL*	33.4453	33.5869
South	31.0179	33.4214	*NM*	2.39284	2.39284
CB	25.986	26.451	*NR*	5.6325	6.0317
CH	54.7174	56.7122	*NW*	61.9436	61.9436
CK	6.72969	6.72969	*SB*	27.9267	29.5487
CM	23.1931	24.5651	*SH*	68.7975	70.1749
CN	3.42615	3.42615	*SM*	5.58638	5.60133
CW	13.8965	13.8965	*SW*	5.13775	5.13775

5.3.2 Restart Intensification

The alternative intensification method was designed to adjust the initial distribution of available staff members before initialization of the main tabu search itself. There is potential for increase to solution quality through this as the preliminary search should result in an improved starting solution for the main tabu search; effectively the second stage of the search will then initiate in a more profitable region of the solution space. This is achieved through conducting a preliminary search that uses only a restricted subset of available elements. In order to identify such elements to be used within the preliminary search information is drawn from a previously conducted optimizing search. In a similar style to the restart diversification method, the restart intensification considers the frequency with which each element of interest was selected for use within the previous search. In this case, with staff distribution being of interest, 2 separate counts are kept for each shift; one count records the amount of times that a staff member is moved to that shift and the other the amount of times a staff member moves from it. In contrast with diversification however the point of interest this time is in discovery of those shifts that are most commonly used, being those with the highest expected benefit either as a donator or receiver of staff members.

With a set of most commonly used elements defined, the preliminary intensification search process may be applied on restarting the search in order to perturb the initial roster before implementing the tabu search process. The preliminary search performs precisely as a local neighbourhood search acting with a restricted set of available moves. By default this restricted move set is defined to allow only those moves in which staff are transferred from previous common donators to previous common receivers. Alternative move set restrictions are also defined through placement of restriction only upon the donating shifts or receiving shifts, these allowing a greater level of diversity within the restart intensification.

5.4 Computer Implementation of Multiple Minima Diversification

As an aid to clarity, the flow chart displayed in Figure 5.1 describes the process of implementation for a tabu search using all of the following described diversification and intensification approaches. This shows not only how multiple minima diversification is incorporated but also restart diversification and restart intensification that are described in more detail within section 5.5.

For this, each stage of the tabu search will maintain a record of the most recently conducted moves towards optimization; for the starting time variation search stages a single line array named 'Move_Times_Record[i]' is used for this purpose, whereas for the staff redistribution search stage an array named 'Move_Dist_Record[j][i] is used instead. In the first of these cases the array records simply the number of the shift that has been subject to starting time perturbation with the value of i indicating the recency of the perturbation. So for example, Move_Times_Record[2] would record the value for the second most recently occurring perturbation. In the second case the first array element (j) indicates whether a staff member has been moved to or from a particular shift; again the second value i indicates te recency of the perturbation. For example Move_Dist_Record[1][i] would indicate that a staff member had

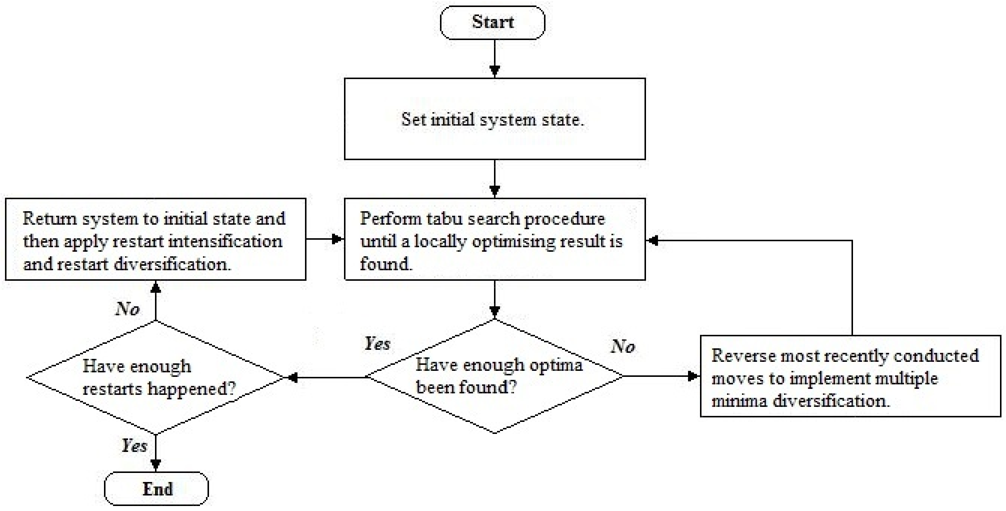
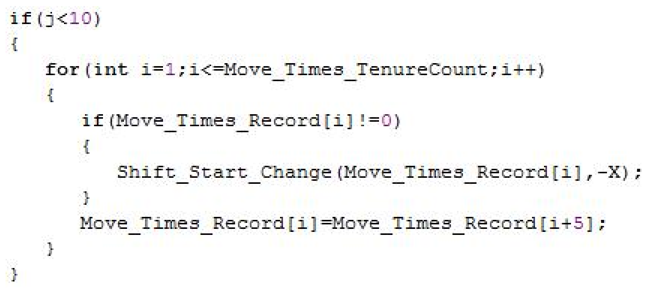

Start
Set initial system state.
Return system to initial state and then apply restart intensification and restart diversification.
Perform tabu search procedure until a locally optimising result is found.
No
Yes
No
Have enough restarts happened?
Have enough optima been found?
Reverse most recently conducted moves to implement multiple minima diversification.
Yes
End

Figure 5.1: Flow chart describing the overall tabu search process with diversification and intensification techniques.

been moved from a shift whereas Move_Dist_Record[2][i] would show that a staff member was moved to a shift.

The multiple minima diversification makes use of these arrays in identifying the most recent of the previously conducted moves in order to allow for their reversal upon discovery of a locally optimizing solution. When such a solution is found code following the steps shown in Figure 5.2 is implemented to cause the move reversal process; the code presented is for the stage of the search at which shift start time increase is investigated. The first if statement is used to check whether a desired number of local minima have yet been discovered, by default this has been set to 10 such that the move reversal process will occur 9 times in total prompting the discovery of 10 local optima. For all non-zero values within the list of moves to be reversed a subroutine named 'Shift_Start_Change()' is called to perform the shift starting time re-adjustment. Upon the reversal of a move, it's record within the Move_Dist_Record array is overwritten by that for the perturbation which occurred 5 moves previously. In this way the most recently occurring 5 moves are reversed and the list of most recent moves is updated such that the 6^{th} through 10^{th} most recent become the 1^{st} through 5^{th}.



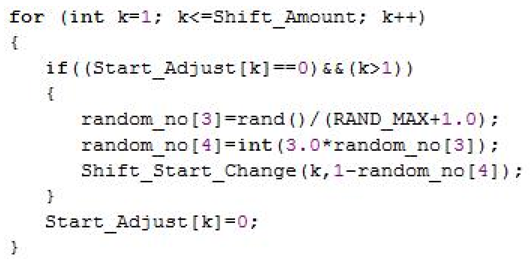
```
if(j<10)
{
   for(int i=1;i<=Move_Times_TenureCount;i++)
   {
      if(Move_Times_Record[i]!=0)
      {
         Shift_Start_Change(Move_Times_Record[i],-X);
      }
      Move_Times_Record[i]=Move_Times_Record[i+5];
   }
}
```

Figure 5.2: C++ code detailing multiple minima diversification implementation.

5.5 Computer Implementation of Restart Diversification and Restart Intensification

As mentioned within Sections 5.2.3 and 5.3.2 one of the key requirements in order to implement a restart diversification or intensification technique is the recording of useful information from a previously conducted search. In order to achieve this a series of 3 arrays are used that record the number of times each of the available shifts is selected for adjustment between iterations of the search; that is, the number of times a move of a given type is applied to each shift. The first of these arrays is named Start_Adjust[i] and (as with the other 2 search data recording arrays) is initially created with each element set to a value of 0. As any shift is selected for starting time adjustment within either of the shift start time redistribution stages of the tabu search process the associated element of the Start_Adjust array will be increased by 1. As each shift is identifiable by an integer value this means that if shift number X is perturbed then Start_Adjust[X] will be increased by 1. The other 2 arrays are named Level_Adjust_To[i] and Level_Adjust_From[i] and are used within the staff redistribution stages of tabu search in order to record the number of times a shift is selected to have a staff member moved to it or from it respectively. For example, if the most beneficial move at a particular iteration of the staff redistribution search is to move a staff member from shift X to shift Y then Level_Adjust_To[Y] and Level_Adjust_From[X] would both be increased by 1 to reflect this.

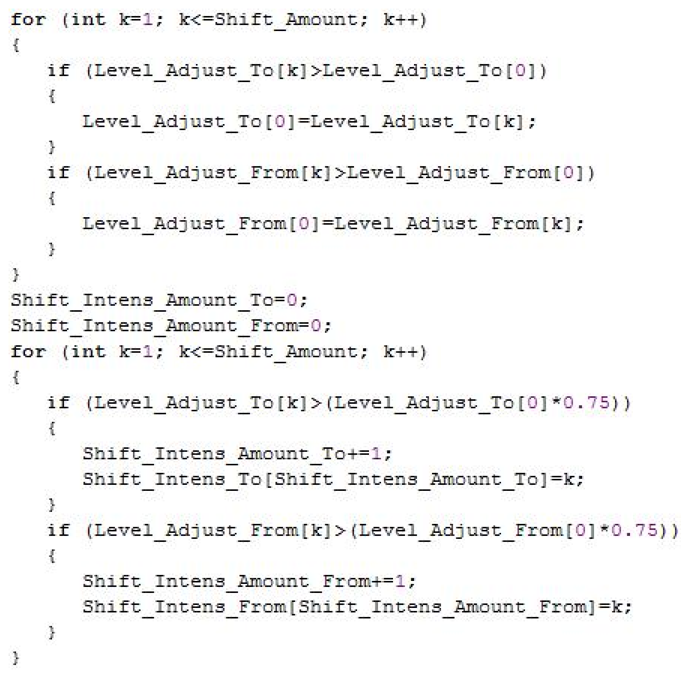
The restart diversification process is contained within a 'Restart_Diversification()' subroutine that makes use of the array Start_Adjust in identifying and perturbing those shifts with starting times that were least adjusted during the previously conducted search. In order to implement this a call to the subroutine is performed after the search has been returned to the initial state, thus providing perturbation to the tarting roster for the next search conducted. The subroutine performs using the code illustrated in Figure 5.3. The if statement identifies whether the currently investigated shift k meets 2 criteria, firstly that the starting time was completely unperturbed (Start_Adjust[k]== 0) and that it is not the first of the defined shifts ($k > 1$). In all of the trial problems defined for use by the various search methodologies attempted in this research, shifts were defined within the input .txt files in chronologically ascending order; due to this the earliest starting shift is always the first numerically recorded within the C++ programs. The second criteria then ensures that the earliest starting shift may not be perturbed at all, preventing it from being allocated a starting time that falls before the start of the time period being investigated.



```
for (int k=1; k<=Shift_Amount; k++)
{
    if((Start_Adjust[k]==0)&&(k>1))
    {
        random_no[3]=rand()/(RAND_MAX+1.0);
        random_no[4]=int(3.0*random_no[3]);
        Shift_Start_Change(k,1-random_no[4]);
    }
    Start_Adjust[k]=0;
}
```

Figure 5.3: C++ code detailing the 'Restart_Diversification()' subroutine.

Restart intensification is also accomplished through a call to a subroutine upon search restart, this time the subroutine being named 'Restart_Intensification()'. This is a more complex process than the restart diversification as it is necessary to perform a preliminary tabu search process as part of the subroutine. The first step of the intensification subroutine is in using the data recorded within the arrays Level_Adjust_To and Level_Adjust_From in identification of those shifts that were the most common donators and receivers of staff within the previously conducted search. To achieve this, first the most commonly used shift of each type are found and then 2 subsets of all available shifts are defined which are comprised of this shift and those that are used nearly as much. The method by which this occurs is illustrated within the C++ code shown in Figure 5.4. The first for loop within this code finds the maximum recorded value within the Level_Adjust arrays and stores them within the previously unused Level_Adjust_To[0] and Level_Adjust_From[0] array entries. With these discovered the second loop creates a subset of all those shifts which were used as donators or receivers of staff an amount of times within some range of the most commonly used. Within the sample shown, an example condition is set such that only those shifts that were used at least 75% of the times that the most commonly used was are considered within the created subsets. This value was selected through consideration of the number of times individual elements were commonly seen to be selected through a tabu search process; it was commonly seen that approximately a third of available shifts would be utilized within the 75% range. From this 2 subsets of shifts are defined that are then used as the exclusive set of shifts from which the preliminary search is conducted; these subsets being Shift_Intens_To[k] and Shift_Intens_From[k]. The preliminary search performs in the same way as the base tabu search method except that it may move staff only from those shifts recorded in the Shift_Intens_From subset and only to those in the Shift_Intens_To subset.



```
for (int k=1; k<=Shift_Amount; k++)
{
   if (Level_Adjust_To[k]>Level_Adjust_To[0])
   {
      Level_Adjust_To[0]=Level_Adjust_To[k];
   }
   if (Level_Adjust_From[k]>Level_Adjust_From[0])
   {
      Level_Adjust_From[0]=Level_Adjust_From[k];
   }
}
Shift_Intens_Amount_To=0;
Shift_Intens_Amount_From=0;
for (int k=1; k<=Shift_Amount; k++)
{
   if (Level_Adjust_To[k]>(Level_Adjust_To[0]*0.75))
   {
      Shift_Intens_Amount_To+=1;
      Shift_Intens_To[Shift_Intens_Amount_To]=k;
   }
   if (Level_Adjust_From[k]>(Level_Adjust_From[0]*0.75))
   {
      Shift_Intens_Amount_From+=1;
      Shift_Intens_From[Shift_Intens_Amount_From]=k;
   }
}
```

Figure 5.4: C++ code detailing the 'Restart_Intensification()' subroutine.

5.6 Results from Diversification and Intensification

With suitable diversification and intensification methods defined for application within the tabu search algorithm it proved suitable to trial their use individually and in collectives in order to assess the impact upon resultant solution quality.

5.6.1 Multiple Minima Diversification Implementation

Being the simplest method to incorporate within the defined search algorithm, the first trials involved inclusion of only the multiple minima diversification alongside the base search approach. There is one key factor that may be varied at the initiation of the defined tabu search method, namely the tabu tenure penalty. In this case and all of those following, the tabu tenure penalty approach used remains the same. The tabu tenure considered is a recency based objective function penalty that directly subtracts the current tenure values associated with any tabu-active elements for each move investigated. For example, a potential move that involves 2 tabu-active elements with recency tenure 4 and 3 respectively would have a penalty of 7 applied to its objective function value. The maximum recency tenure applied for each move is 5, as such any element involved in a move will be marked with a tenure of 5 with this value decreasing as further search iterations occur.

Upon application of the multiple minima diversification to each of the trial problems, a set of comparisons were made between searches terminating after finding only the first local optima and ones using the multiple optima search method as described. The multiple minima search process was set to allow the algorithm to find up to 10 unique locally optimizing results for the searches involved for each move type. The results of this are displayed in Table 5.2 for each of the 18 test problems. Within this figure, the columns titled 'Multiple' indicate the best roster scores found using multiple minima diversification with tabu search and the column titled 'Single' show best roster scores from using just tabu search with no intensification or diversification techniques. This data would seem to suggest that the multiple optima search process was of no benefit at all within the 18 test problems; indeed this result indicates that no improvement to the best roster has been made beyond the first optimum found.

Table 5.2: Comparison of tabu search results using single minima and multiple minima search procedures.

Region	Multiple	Single	Region	Multiple	Single
City	39.784	39.784	*NH*	20.3039	20.3039
North	31.7311	31.7311	*NL*	32.7987	32.7987
South	30.3353	30.3353	*NM*	2.39284	2.39284
CB	26.3009	26.3009	*NR*	6.11011	6.11011
CH	55.8845	55.8845	*NW*	61.9436	61.9436
CK	6.72969	6.72969	*SB*	27.9267	27.9267
CM	25.3516	25.3516	*SH*	68.7975	68.7975
CN	3.6575	3.6575	*SM*	4.7377	4.7377
CW	14.954	14.954	*SW*	5.23825	5.23825

Effectively this means that each of the further locally optimizing solutions discovered are worse than the first for each of the trial problems considered. However the inclusion of this stage does at least ensure that the search results provided are optimal within a larger region of the search space; up to 9 more nearby optima have been found in each case that provide no improvement to the best found solution. For example, the solution quality values of all 10 optima discovered when applied to the City region problem are illustrated in Table 5.3

Table 5.3: Objective function values for each of the 10 local optima found using multiple minima search on the City region problem.

Optimum	Objective Function Score
1	39.784
2	40.326
3	42.135
4	41.287
5	39.920
6	40.158
7	39.784
8	41.115
9	43.217
10	40.892

It would appear that the 9 local optima discovered through multiple minima diversification in this example provide no improvement to the quality of the first solution found in this example. In this case it may be stated that the first found solution is the strongest available within some sub-region of the solution space and a stronger diversifying perturbation would be required in order to reach an area of the solution space where a stronger solution may be found. One method of achieving such a perturbation within the search is through the use of restart diversification.

5.6.2 Restart Diversification Implementation

The restart diversification approach was next tested in conjunction with multiple minima diversification, however the metric by which success of the method would be measured was also changed. Due to the semi-random nature of restart diversification it is possible that any search for which it is used within the initialization process may perform better or worse than another. Thus it was more suitable to consider the rate of success provided by inclusion of restart diversification, that is the percentage of searches in which it provides an improvement to solution quality over that obtained without it. To calculate such a ratio 10000 repetitions of the tabu search were performed upon a single trial problem, with each repetition being used in order to inform a restart diversification applied to the next. It may at first appear that 10000 searches is a very large amount to conduct in generation of such a success rate figure, however there are approximately $5.003 * 10^{16}$ possible post-diversification search starting points. It is through the compilation of data from a previously conducted search that suitable shifts for adjustment are chosen and 1 of these many starting points is intelligently generated. The trial problem selected for use was that based upon data collected for the City

BCU region of Leicestershire as this provided the most flexible problem with the greatest scope for solution variation. The restart diversification itself acted as described in Section 5.2.3 with least used shifts in a previous search being defined as those that are unused for the entirety of the search. Table 5.4 describes results obtained through a series of 4 trial runs, each consisting of 10000 tabu search repetitions with restart diversification; 4 trials were used in order to provide the ability to spot potential outliers within the results.

Table 5.4: Rates of solution improvement seen using several trials of basic restart diversification with tabu search.

Run	Improvements - (Rate)	Best Roster Score
1	1740 - (17.4%)	24.9836
2	1704 - (17.0%)	23.4397
3	1647 - (16.5%)	24.8579
4	1729 - (17.3%)	23.4612
Average	(17.1%)	

It should be noted that such an increase in the amount of searches conducted in order to generate a final result had an effect upon the total calculation time required; indeed this is as anticipated with 10000 full searches being conducted for each run. The average time for total run completion rose from a couple of seconds up to the region of 20 minutes. The restart diversification for the first runs was limited to only allow adjustment of the defined shift starting times by up to a single hour; in effect this meant that a shift starting time selected for perturbation could (through random selection) be increased by 1, decreased by 1 or left unaltered. The same restart diversification process was conducted upon each of the 15 LPU based trial problems in order to provide further depth to the generated results and to investigate for any results of note. Summary results from these conducted searches are included in Table 5.5. These results display the average rate of improvement and best roster score found during 4 searches; again this number being performed in order to allow for identification of any anomalous results.

To follow this, another series of 4 runs were conducted with a more powerful restart diversification that was able to perturb shift starting times with an increase or decrease of up to 2 hours. This change being performed to encourage a wider region of the potential solution space to be investigated by perturbing the initial roster a greater amount upon each restart of the search. Results provided from these runs are described within Table 5.6.

A notable decrease in the rate at which improving solutions are found is seen when allowing

Table 5.5: Rates of solution improvement across 15 LPU based trials of basic restart diversification with tabu search.

LPU	Improvements	Best Roster Score	LPU	Improvements	Best Roster Score
CB	32.8%	22.0966	NM	4.4%	1.82343
CH	30%	52.11775	NR	54.8%	4.295985
CK	24.6%	5.04691	NW	17.8%	60.14715
CM	75.6%	20.73255	SB	42.4%	23.70565
CN	0%	3.6575	SH	75.8%	62.7177
CW	26.1%	12.20175	SM	18.8%	2.52923
NH	38.7%	17.14185	SW	25.7%	4.2654
NL	76%	27.36565			

Table 5.6: Rates of solution improvement seen using several trials of more powerful restart diversification with tabu search.

Run	Improvements - (Rate)	Best Roster Score
1	341 - (3.4%)	27.3052
2	348 - (3.5%)	24.4409
3	350 - (3.5%)	23.8451
4	334 - (3.3%)	27.2197
Average	(3.4%)	

the diversification a stronger perturbation ability. This suggests that the increased strength of perturbation will commonly result in weaker candidate solution. Additionally from this it can be stated that it is highly likely that a globally optimizing result exists within the near vicinity of the initially defined solution, with large perturbation from this starting point forcing the search away from such an optimum. For comparison, sample solution rosters provided through implementation of restart diversification using both the wider and narrower range of element perturbation values are illustrated in Figure 5.5. Within the graph shown the shaded region is indicative of the expected demand level for the trial problem with the 2 lines imposed on top of this indicating staff allocations suggested by generated rosters. The complete line indicates the staff roster associated with the tabu search using restart diversification perturbing selected elements by only a single hour, whereas the dashed line indicates the staffing roster resulting from tabu search with restart element perturbation of up to 2 hours. Of note, the region of the graph that lies between 25 and 33 hours indicates a period over which the wider diversification has resulted in a large amount of understaffing. This indicates that the stronger element perturbation does indeed lead to situations that are not beneficial to the search, explaining the lower rate of solution improvement it was shown to provide.

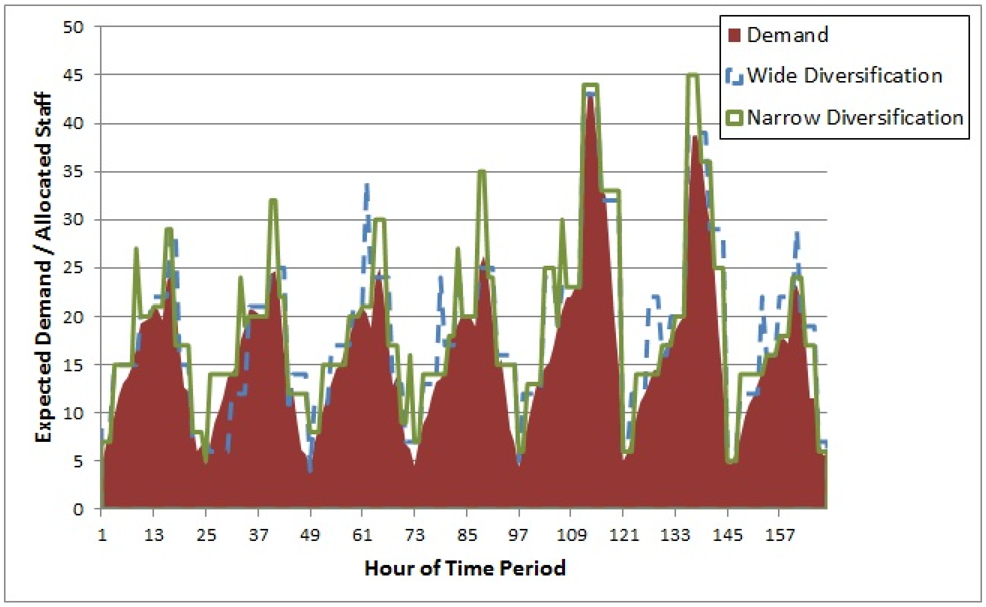


Figure 5.5: Comparison of rosters resulting from tabu search with variation in restart diversification parameters.

It was necessary to verify this result by comparison over a wider range of problems, in this case the modified restart diversification search was conducted upon the 15 LPU based trial problems. Results from this are displayed within Table 5.7 with the percentage rate of improvement and best found roster score averages from 4 searches upon each problem. These results may be directly compared with those contained within Table 5.5 to allow for validation of the claim that the more powerful restart diversification that is able to perturb the initial roster to a greater degree is in fact of detriment to the search. This may be seen through the fact that for each of the defined LPU trial problems the weaker restart diversification outperforms the more powerful one; except in the single case given by the CN trial problem, in which neither search method is able to improve solution quality.

For the restart diversification method applied in this work, adjustments were made only to the first defined starting roster. This means that the initialization point before application of diversification or intensification was the same between each series of runs and between each search conducted within a run. The effect of this is to restrict the ability of restart diversification in selecting shift starting times that are too far from what may be feasible to actually be implemented within a real staffing roster. One method by which this restriction may be removed is through preserving the changes that are made to the search initialization point through restart diversification. In this way the initialization point will vary more

Table 5.7: Rates of solution improvement across 15 LPU based trials of more powerful restart diversification with tabu search.

LPU	Improvements	Best Roster Score	LPU	Improvements	Best Roster Score
CB	11.2%	22.6762	NM	2.4%	2.02046
CH	10.1%	52.77405	NR	34.2%	4.02388
CK	1.7%	5.69582	NW	4.2%	60.0717
CM	47.5%	20.757	SB	33.2%	23.2752
CN	0%	3.6575	SH	44.1%	63.9632
CW	5.5%	12.6112	SM	9%	2.99468
NH	17.2%	17.2155	SW	11.9%	4.05338
NL	60.2%	26.7712			

widely between searches and individual shifts may eventually become perturbed a great distance from their very first position. For example, consider Tables 5.8a and 5.8b. Table 5.8a could be representative of a staffing roster that is defined as the original search starting solution, it is this solution that will always form the pre-diversification starting point in the default methodology used. If Table 5.8b represents the post-diversification point of the second search, then in the alternative method mentioned this will become the pre-diversification state for the third search. As restart diversification considers only the perturbation of shift starting times then in this example it is only these values that change; 09:00 becomes 10:00, 14:00 becomes 13:00 and 17:00 becomes 18:00.

Table 5.8: Sample initial rosters for use in tabu search.

Start	Duration	Staff
07:00	9	6
09:00	9	0
14:00	9	7
17:00	9	0
22:00	9	6

(a)

Start	Duration	Staff
07:00	9	6
10:00	9	0
13:00	9	7
18:00	9	0
22:00	9	6

(b)

Performing such restriction alleviation however resulted in tabu searches that were unable to provide improvement to solution quality beyond that gained without restart diversification; through several runs of 10000 searches, not a single improving result was found. Indeed the high level of shift variability was seen to result in solutions that were of particularly low quality due to shifts being commonly redefined in such a way that regions of the target demand profile could not be supplied with staff at all. This was a common problem seen in all of the defined trial problems and indeed not a single positive result was generated from searches conducted using this adjustment.

5.6.3 Restart Intensification Implementation

Restart intensification was applied to the problem in conjunction with restart diversification and multiple minima diversification in 3 different ways. The first method attempted was to use the full restart intensification method as described in Section 5.3.2 wherein the most frequently used elements are identified and a tabu search making exclusive use of these elements is performed as part of each search initialization. The 2 further methods are defined through preliminary searches that are less restrictive on the available element set, invalidating fewer elements as available choices within the search. Such lighter restriction is achieved by placing limitations only upon either the elements from which a staff member may be moved from or those which a staff member may move to. Similarly to restart diversification application, the objective used in order to assess the ability to impact upon solution quality of restart intensification was calculation of an improvement rate and recording of best solution quality over a set number of system restarts; the number of restarts selected once again being 10000. The results detailing solution quality improvement rates for inclusion of the restart intensification process on the City BCU trial problem are summarized in Table 5.6.3.

Full Intensification	Limit Intensification (to)	Limit Intensification (from)
1681 - (16.8%)	1810 - (18.1%)	1659 - (16.6%)
1752 - (17.5%)	1643 - (16.4%)	1631 - (16.3%)
1732 - (17.3%)	1928 - (19.3%)	1693 - (16.9%)
1724 - (17.2%)	1762 - (17.6%)	1572 - (15.7%)
17.2%	17.9%	16.4%

It is seen from this that addition of full restart intensification on top of diversification approaches yields no significantly greater level of improvement with regards to rate of solution improvement (a change from 17.1% to 17.2%). A much more notable change occurs through use of either of the more restricted restart intensification approaches. Average rate of solution improvement is raised through inclusion of restriction only upon the shifts which staff members may be moved to during the preliminary search. Whereas limitation only upon which shifts staff may be moved from within the preliminary search is seen to be counterproductive, worsening the overall rate of solution improvement. In order to fully explore the potential of this restart intensification method a further series of trials were conducted upon the 15 LPU based trial problems, with restriction during restart placed only upon the shifts which staff members may move to. Results of this are shown in Table 5.9. Comparison of these results with those in Table 5.5 shows that in 9 out of the 15 trial problems restart intensification improves the rate at which the search is able to discover improving solutions. Notably, application of restart intensification is the first case in which the tabu search shows an improvement to solution quality beyond that given by the base tabu search for the trial problem relating to the CN LPU.

Table 5.9: Improvement rates from use of restart intensification on 15 LPU based trial problems.

LPU	*Rate*	*LPU*	*Rate*
CB	42.2%	*NM*	4.2%
CH	26.4%	*NR*	60.4%
CK	13.3%	*NW*	18.9%
CM	86.6%	*SB*	44.2%
CN	0.4%	*SH*	75.5%
CW	39.4%	*SM*	26.3%
NH	34.9%	*SW*	28.1%
NL	92.4%		

5.6.4 Summary

At this point, the best performing tabu search has been found to make use of the following key factors:

- Following a process by which moves are conducted in sequence until an optimum is found. This sequence is staff redistribution, followed by shift starting time increase, then shift starting time decrease and finally a secondary staff redistribution.

- Upon discovery of an optimum, multiple minima diversification perturbs the search away from this result and the search continues.

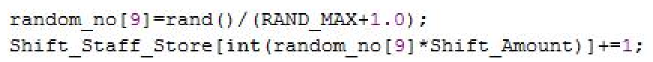
- Application of restart diversification in order to perturb the shift starting times of the initial system state.

- Use of restart intensification to conduct a preliminary tabu search focusing on a specific subregion of the potential solution space.

5.7 Adaptive Total Staffing Level

All searches conducted up until this point have considered the same objective function, that being the total amount of expected demand that remains unmet by the generated staffing roster. It was noted that by the definition of the theoretical problems used for trials it was impossible for a resulting roster to be possible to meet all such demand.

A new objective was defined in which the total number of staff members available for rostering is increased incrementally in order to find the minimum number of staff members required in order to meet all of the expected demand; a search terminating upon discovery of a first solution with quality of 0 as measured by the original objective function. Importantly these increases in total available staffing occur between full tabu search implementations and as such the best possible arrangement of available staff is provided for each total staffing level investigated. Due to this, the first discovered solution with an objective function value of 0 should be one that requires the fewest total number of staff. For example, the tabu search will perform with X amount of staffing members available for allocation, upon termination and restart it will then consider $X+1$ staff available. Restart diversification was applied as described in Section 5.2.3 with a range of $\{-1, 0, 1\}$ for potential perturbations; other methods were not chosen as they had been identified as little if any impact on solution quality. The base idea behind the adaptive total staffing method is to first perform a full tabu search with a set number of restart diversifications and then to perturb the system in such a way that further searches will be able to find other optimizing solutions. This perturbation is achieved through the addition of a single member of staff to a randomly selected shift within the initial staffing roster. Code written within C++ in order to accomplish this is included in Figure 5.6. An array is used in order to record the total number of staff within the initial roster, namely 'Shift_Staff_Store[i]', increasing an element of this array then has the effect of adding an additional staff member to one of the shifts within te starting roster. The code demonstrated in Figure 5.6 randomly selects one of the shifts within the starting roster and increases the number of staff initially assigned to it by 1. This code is contained within an overall system loop that performs until a solution is found whereby all demand is met; the search will then continue to increase staff until a solution that meets all demand is found.



```
random_no[9]=rand()/(RAND_MAX+1.0);
Shift_Staff_Store[int(random_no[9]*Shift_Amount)]+=1;
```

Figure 5.6: C++ code describing basic adaptive total staffing level search.

Through this, the adaptive total staffing level search algorithm will find an optimizing solution for the initially defined number of staff members and will then proceed to find similar optima for enhanced staffing levels until a solution is found that meets all demand, at which point it will terminate. This process may be summarized by the flow chart presented in

Figure 5.7.

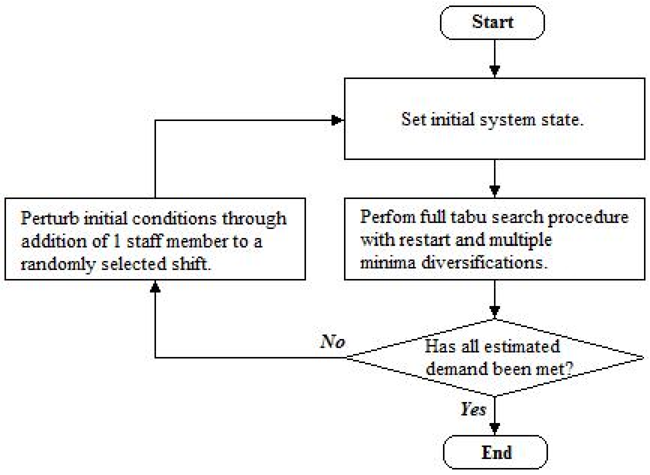

Start
Set initial system state.
Perturb initial conditions through addition of 1 staff member to a randomly selected shift.
Perfom full tabu search procedure with restart and multiple minima diversifications.
No
Has all estimated demand been met?
Yes
End

Figure 5.7: Flow chart describing the adaptive total staffing level process.

As all resulting best found rosters when performing a tabu search with this additional component will attain an objective function value of 0, a new objective is required in order to directly compare their relative strengths. This can be achieved through consideration of the number of additional staff members required by the search in order to generate a roster capable of meeting all demand; this number being directly comparable to the total number of staff used within the final roster. Indeed if 2 or more rosters are presented that each meet all simulated demand but with different total staffing levels, then it would be intuitive to use the roster that would cost the least to implement. Note that this method of assessing the effective cost of a roster is only suited in this case as each of the shifts is of the same duration. In a case where the shift length could be seen to vary from one shift to the next then a more detailed calculation would be required to take into account the total amount of work hours assigned instead of total number of shifts.

In order to assess the capability of the adaptive total staffing level search addition, a series of trial simulations were conducted using varying total amounts of restart diversifications per search. In this way it was intended to obtain an overall indicator of the strength of total staff level variation whilst also investigating the effect (if any) of restart diversification. A potential benefit to be gained would be the identification of some upper limit on the total number of restarts beyond which no further enhancement to final solution quality is commonly seen. This information would allow future searches to be conducted using only the discovered number of restarts or fewer, avoiding unprofitable calculation and the associated additional computation time.

Table 5.10 describes the results from these trial simulations through provision of the total number of additional staff members allocated through the search in order to meet all predicted demand. The variation that exists within each of the tabu restart quantity categories is present due to the semi random nature by which restart diversification is applied; this is through random perturbation of shift starting times at the initialization of a search.

Table 5.10: Total additional staff required to meet demand using basic adaptive total staffing level.

Run	25 Restarts	50 Restarts	75 Restarts
1	53	40	52
2	53	52	51
3	54	53	54
4	45	52	52
Average	51.25	49.25	52.25

Creating a graphical representation of the final generated solutions however it became immediately clear that the search was underperforming, with regions of the time period considered experiencing a surplus of staff availability. For clarification, an example visualization used with regions of overstaffing indicated by peaks in supply is provided in Figure 5.8.

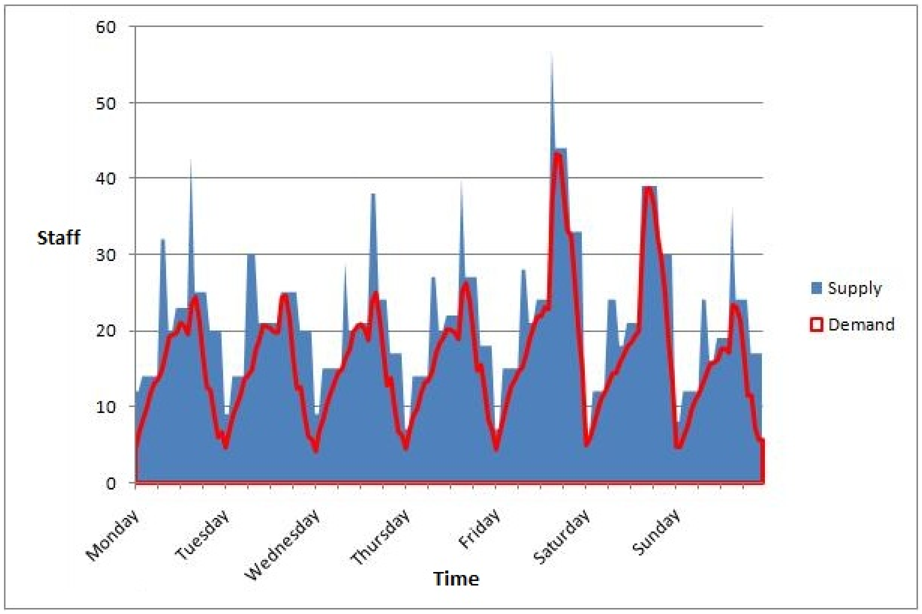
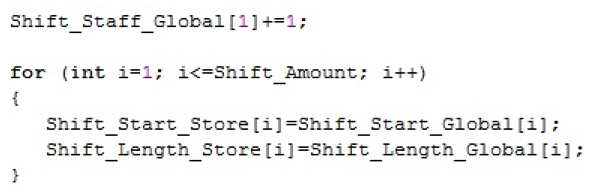


Figure 5.8: Graphical output from basic adaptive staffing level search.

The results provided in Table 5.10 are indicative that increasing the number of system restarts used bears no significant impact upon the quality of final solutions provided by the search algorithm. From this it would be expected that the most effective choice for total number of system restarts would be the lowest, 25. This also suggests that restart diversification has no strong benefit to the new objective. However with this total staff adjustment process, the search will be reset to the same initial condition between each of the overall tabu searches conducted with the addition of a single staff member to a single shift. Due to this any of the benefit discovered through conducting the previous search will be lost and will bear no effect at all upon the development of future searches. Thus it is likely that improvement to performance would be seen through retention of some characteristic from previously conducted searches when performing a new one.

The adaptive total staffing level search was seen to perturb the system through addition of staff members to a randomly selected shift of the initial starting roster before restarting the entire search. This is not ideal practice for 2 major reasons. The first of these is that in forcing the search to initiate from the same point each time there is a lack of search diversity as the same solution space region is investigated repeatedly. The second reason is the fact that it would be expected that a similar path to optimization would be taken from such a similar initial system state and thus a reasonable amount of calculation time could be expected to be used in the repetition of moves conducted in a previous search.

In an attempt to cater for this, a modification to the search process was created in which the best roster found during a full tabu search was used as the pre-diversification starting point of the following search upon system restart. Thus the next search process would seek to improve upon the best previous solution through perturbation of shift starting times followed by the standard tabu search procedure. Importantly only the shift starting times and durations are retained, with staffing levels being reset to the original total of 350 staff members. The search will use these shifts as a basis and once more attempt incremental increase in staffing until a roster is generated that meets all demand. This approach should still allow the search a good degree of diversity whilst encouraging it to also focus on an area of the solution space with strong potential for high quality results. This is achieved through use of a segment of C++ code as illustrated within Figure 5.9. Here a series of 'Global' arrays are defined which retain the current best found roster as the search progresses, these arrays are used to update the 'Store' arrays which govern the initial system state. Placement of this code after a search has completed will result in the overwriting of the original stored system state with the current best found optimizing solution.



```
Shift_Staff_Global[1]+=1;

for (int i=1; i<=Shift_Amount; i++)
{
   Shift_Start_Store[i]=Shift_Start_Global[i];
   Shift_Length_Store[i]=Shift_Length_Global[i];
}
```

Figure 5.9: C++ code detailing the best roster retention mechanism used within adaptive total staffing search.

It may appear at first that this methodology is far more restrictive in the potential solution space to be investigated; with an optimizing solution, addition of a single staff member could be anticipated to only experience freedom of placement with that single member. However through the inclusion of restart diversification such restriction is not a strong factor. Restart diversification is applied through adjustment of shift starting times upon the initialization of a search. This will ensure that the initial state will be perturbed away from the previously found optimizing solution through this diversification at each search restart.

The same series of tests as performed for the base adaptive total staffing level were also conducted in order not only to assess the strength of the method and restart diversification but also to allow direct comparison to be drawn between the methods with and without roster retention. The results of applying this best roster retention addition are displayed in Table 5.11 where it is immediately apparent that inclusion of best roster retention has greatly reduced the total additional staff required to meet demand. Indeed it is seen that the average additional staff required decreases by approximately 25 staff members.

Table 5.11: Total additional staff required to meet demand using adaptive total staffing level and best roster retention.

Run	25 Restarts	50 Restarts	75 Restarts
1	25	23	21
2	19	24	29
3	24	24	19
4	26	26	28
Average	23.5	24.25	23

This result could also be visualized with a graph displaying supply against demand from a sample result, Figure 5.10 illustrating this for the first search run using 75 system restarts. A direct comparison between this result and that shown in Figure 5.8 may be drawn as it is seen that the resulting staffing profile is smoother, with a desired reduction in the sharp peaks of overstaffing seen. However there are still a few areas shown in Figure 5.10 in which overstaffing occurs. These exist due to overlaps between shifts, effectively meaning that several shifts are active at a particular point in time. It would be possible to remove these peaks through reduction of the duration of any of the shifts involved in creation of such a peak; either by ending a shift earlier, or starting a different shift later. To this end a new method was required that allowed for perturbation to the durations of each of the defined shifts used within the search.

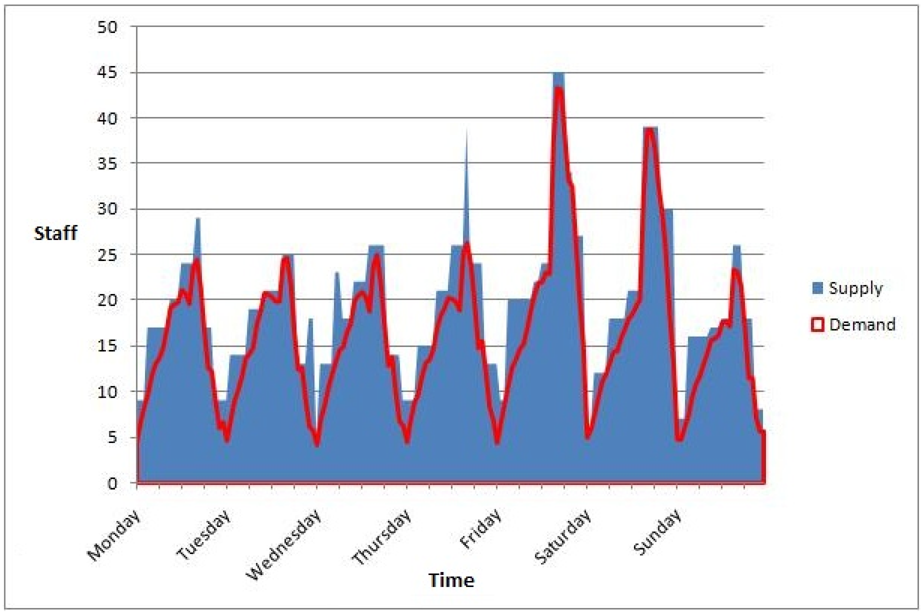


Figure 5.10: Graphical output from adaptive staffing level search with best roster retention.

5.8 Adaptive Total Staffing Level with Semi-Random Shift Length Variation

In order to further increase the capability of the tabu search used within this research it was important to consider other factors that may be automatically perturbed through the search in order to achieve optimal solutions. The only other major factor is the length of a defined shift within a roster. A first simple mechanism by which the duration of shifts may be varied within the search was accomplished through the use of a random perturbation. Similar to the restart diversification method this perturbation was implemented upon completion of a tabu search process, during the initialization process of a further search. In order to perturb the duration of shifts in this stage of initialization a series of shifts are randomly selected to have their length increased by 1 hour and another series of shifts are selected to have their length reduced by 1 hour. Definition of shifts as a member of either series occurs through random selection with the chance of selection for each individual shift of 5%. Several different selection rate values were tested and it was seen that the best performing was 5%; indeed selection rates of higher than 15% were seen to be highly detrimental to final solution quality. Table 5.12 displays results obtained through trials using a variety of selection rates, displayed are the selection rate trialed alongside the average amount of additional staff required in order to meet all demand across 4 trials. It is therefore possible that a single shift could indeed become a member of both series of shifts resulting in both an increase and decrease in shift duration of 1 hour, thus effectively experiencing no adjustment; rate of occurrence of such an event would be anticipated to be 0.25%. With rate of shift selection for duration perturbation set at 5% it would be expected that the majority of shifts would remain unperturbed. In the scenario where shift duration changes are not preserved between searches this would result in the situation of most shifts retaining duration of 9 hours. This may be compensated for by increasing the selection rate in order to allow a wider search of the potential solution space.

Table 5.12: Total additional staff required to meet demand using a range of shift length perturbation selection rates with adaptive total staffing.

Selection Rate	Additional Staff
2.5%	27.25
5%	25.75
10%	26.75
15%	28.5
20%	35.25

An important consideration with shift length adjustment was the coupling with retention of previous best found rosters between tabu searches. Retention of all data from a previously discovered roster entails that the shift starting time, staffing allocation and duration are all preserved. With the preservation of prior best shift durations it is possible for a future tabu search to perturb these durations further from their initial values. In some scenarios where strong variation in shift duration is not a concern this may indeed prove to be an ideal behaviour. With application to the Police staffing problem as posed by Leicestershire constabulary however, it is far more suitable that shifts only be allowed to vary between 8 and 10 hours in length; these being the most commonly assigned shift lengths. There are two modifications to be used in conjunction that allow this restricted variation to be achieved. First of these is limitation of the data retained between tabu searches to comprise only the shift starting times and their associated staffing levels. Second is through the definition of an initial roster state that contains only shifts of 9 hours in duration. Making these adjustments and thus enforcing shift duration to vary only between 8 and 10 hours, a series of trial optimizing searches were conducted; the results from these trials being displayed within Table 5.13.

Table 5.13: Total additional staff required to meet demand using adaptive total staffing level and shift length variation.

Run	25 Restarts	50 Restarts	75 Restarts
1	24	26	22
2	28	25	21
3	23	28	28
4	24	25	22
Average	24.75	26	23.25

For completeness and to provide a fuller picture of the effectiveness of this adaptive staffing level approach, the same technique was applied to the 15 LPU based trial problems using 50 search restarts. With no strong benefit seen through any particular amount of restarts above any other, 50 was as suited in this role as any other of the figures provided; 25 was not selected on the possibility that it would restrict the investigate region of solution space in other problems. The minimum number of additional staff members required in order to generate a solution meeting all demand in each case is presented in Table 5.14.

Table 5.14: Additional staff required to meet demand using adaptive total staffing level and semi-random shift length variation on 15 LPU based trial problems.

LPU	Staff	LPU	Staff
CB	22	NM	9
CK	28	NR	7
CH	15	NW	26
CN	18	SB	19
CM	12	SH	24
CW	15	SM	10
NH	18	SW	10
NL	22		

The main strengths of this method are the simplicity with which it may be implemented and the relatively low calculation cost required for its use. With a random method however there is no strength of design or intelligence to the selection process. It would be far preferable to use a method whereby either some specific objective may be achieved or some particular planned piece of information may be obtained through its inclusion.

Further, there is potential through application of this method in a search experiencing full tabu searches that result in no improvement being found to the current globally best found score. The issue occurs in cases where shifts are randomly selected to experience a reduction in duration. In such cases it may become impossible for the newly redefined shift structure to provide an improvement to the current best scoring roster; this being true even with the increment in total staffing level. The most notable example of such an occurrence would be the situation in which reduction of duration of a single shift results in a roster in which a period of time exists which is covered by no shifts whatsoever. Such a situation could likely be resolved through automated shift starting time perturbations; however this would almost certainly provide a reduction in roster quality and not yield a suitable candidate to replace the current best found solution. Indeed, this precise problem was seen to occur frequently through trial optimizing searches in which best roster shift durations were preserved between tabu searches along with best shift starting times and staff levels. It was common to see final rosters resulting from trial searches requiring approximately twice the number of additional staffing members of those found when using only restricted roster retention. The simplest way in which to overcome this is to limit the adjustment of shift starting times through random selection to increase only; though this would then necessitate further search algorithm change in order to prevent increase in shifts beyond reasonable limits. This further strengthens the appeal of a search algorithm mechanism wherein a more controlled or systematic approach is taken as regards adjustment to individual shift duration.

5.9 Adaptive Total Staffing Level with Methodical Shift Length Variation

A methodical shift duration perturbation was constructed through first enforcing that shift duration adjustment through the random selection process could only cause an increase of 1 hour and could never cause a decrease. This alone would not be suitable as it could lead to a situation in which best rosters are constructed containing shifts of unrealistically large duration. This would only be the case in searches in which shift length is preserved between iterative tabu searches and not so where instead the restricted dataset of shift starting times and staffing allocation are preserved. Indeed the preservation of only best found shift starting times and staffing allocations would ensure that shift durations are reset to their initial default values before the conduction of a tabu search; allowing shifts to only increase by a maximum of 1 hour in duration at any point throughout the entire optimization process. Preservation of best found shift durations between tabu searches does however allow a wider range of shift lengths to be considered by the search and as such it would be beneficial to create a method whereby feasible results may be obtained that allow such considerations. In order to achieve this, a new stage is added to the tabu search procedure that attempts to reduce shift durations; whether they have been previously elevated through restart diversification or not. This technique acts by attempting reduction in duration of all shifts present in any final best found solution resulting from a tabu search. This stage initiates at the conclusion of a tabu search and selects each shift in turn, beginning with the earliest starting and then proceeding through the rest in order of starting time. Upon selection of a shift, reduction of its duration is attempted. Reduction occurs through an iterative process of shortening the selected shift by a single hour and then recalculating the objective function value score in order to assess whether the reduction has been detrimental to the solution quality. In the case that objective function values worsen as a result of shift length reduction this indicates that a previously well catered section of the demand profile has become understaffed. If objective function value does not decrease after shift length reduction then this means that the roster has been redefined to meet defined demand equally as well whilst using fewer staff hours.

As with other tests performed investigating variation in shift lengths, the objective function used in order to assess search quality is the total number of additional staff required by the final best found roster. A first series of trials were conducted, retaining only shift starting times and staffing allocations from best found rosters between each full optimizing search; the results from which are displayed in able 5.15. There is a suggestion from these results that using a larger amount of system restarts lends itself towards better average solution quality; however due to the randomized elements within each conducted search it is possible that this occurs only through chance. Results when using fewer restarts are not significantly different from the results of previously conducted trials used to test other shift length variation methods. Thus performing a larger number of trial searches with 75 system restarts would aid in identifying whether the result shown in Table 5.15 for this number of restarts is anomalous.

Thus 6 more trials were conducted using the same search attribute specifications. This resulted in a total of 10 results for the tabu search including methodical shift length variation with 75 system restarts and retaining only shift starting time and staffing level between each restart. The total additional numbers of staff members required within the final best found

Table 5.15: Total additional staff required to meet demand using adaptive total staffing level and methodical shift length variation.

Run	25 Restarts	50 Restarts	75 Restarts
1	25	23	17
2	24	26	22
3	26	24	26
4	20	26	18
Average	23.75	24.75	20.75

roster in order to meet all demand for each of the 10 trial optimizing searches conducted are displayed in Table 5.16. This suggests that the values of 17 and 18, though very low, are not outside those that may be expected from the search; such low results are not replicated within the additionally conducted searches and it may be stated that it is not indicative of a new distinct result behaviour.

Table 5.16: Additional results using 75 system restarts with adaptive total staffing level and methodical shift length variation.

Trial 1	Trial 2	Trial 3	Trial 4	Trial 5	Trial 6	Trial 7	Trial 8	Trial 9	Trial 10
17	22	26	18	25	21	28	20	24	25

The tabu search using methodical adjustment to shift lengths was conducted with all trial optimizing searches being re-performed using full roster retention between searches. This means that shift durations were preserved for each shift of the best found roster for the initialization point of the next conducted search in addition to the shift starting times and staffing levels. The results from each of these re-performed searches are contained within Table 5.17. It is apparent from this that those searches in which a larger number of restarts are performed at each iterative stage of each individual tabu search are able to provide final rosters that require lower amounts of additional staff members in order to meet all estimated demand.

It is worth noting however that in the case of full roster retention, it is likely that shifts will remain at elevated duration between tabu search iterations; indeed with full retention there is also the possibility that shifts may attain duration above 10 hours. Accordingly then, a final roster that is seen to require fewer additional staff members than those produced using previous searches, may actually entail a larger total number of man hours. For example the first trial conducted using 50 restarts per iterative tabu search process using full roster

Table 5.17: Total additional staff required to meet demand using adaptive total staffing level with full best roster retention.

Run	25 Restarts	50 Restarts	75 Restarts
1	27	16	19
2	19	19	15
3	26	12	12
4	24	19	16
Average	23.33	16.5	15.5

retention provided a final roster with 16 additional staff members and totalling 3348 man hours across all shifts. In contrast the first trial conducted with the same number of restarts with restricted roster retention gave a final roster using 23 additional staff members with a total of 3362 man hours. Even though the search using full roster retention provides a final best roster requiring 7 fewer staff members, it has only saved 14 man hours. One could have wrongly assumed that a reduction of 7 staff members was equivalent to approximately 63 man hours; this being the amount of hours worked by 7 staff members each on a standard 9 hour shift. This illustrates that an important distinction exists between results generated using methodical shift length variation with full roster retention and other prior tabu search methods within this research. Indeed, a fuller comparison between the methodical and semi-random shift length variation procedures is provided within Table 5.18 where results from application of both techniques across the 15 LPU trial problems are displayed. It can be seen that the methodical approach to shift length variation outperforms (or at least matches the performance of) the semi-random shift length variation approach in all cases.

Table 5.18: Additional staff required to meet demand using adaptive total staffing level and semi-random shift length variation on 15 LPU based trial problems.

LPU	Random	Methodical	LPU	Random	Methodical
CB	22	19	NM	9	6
CK	28	26	NR	7	6
CH	15	12	NW	26	23
CN	18	16	SB	19	19
CM	12	10	SH	24	22
CW	15	13	SM	10	9
NH	18	17	SW	10	8
NL	22	19			

To further illustrate this Figure 5.11 displays results of application of methodical shift length variation alongside random shift length variation upon the same trial problem. The dashed line indicates the staffing level associated with the best found solution using methodical shift length variation and the full line displays staffing level from random shift length variation. There is a clear difference shown, with random shift length variation resulting in a greater scale of variation seen across the roster with significantly larger peaks and deeper troughs in supply. It is due to these inefficiencies and the compensation required that the random shift length variation will commonly be shown to require a greater amount of additional staff.

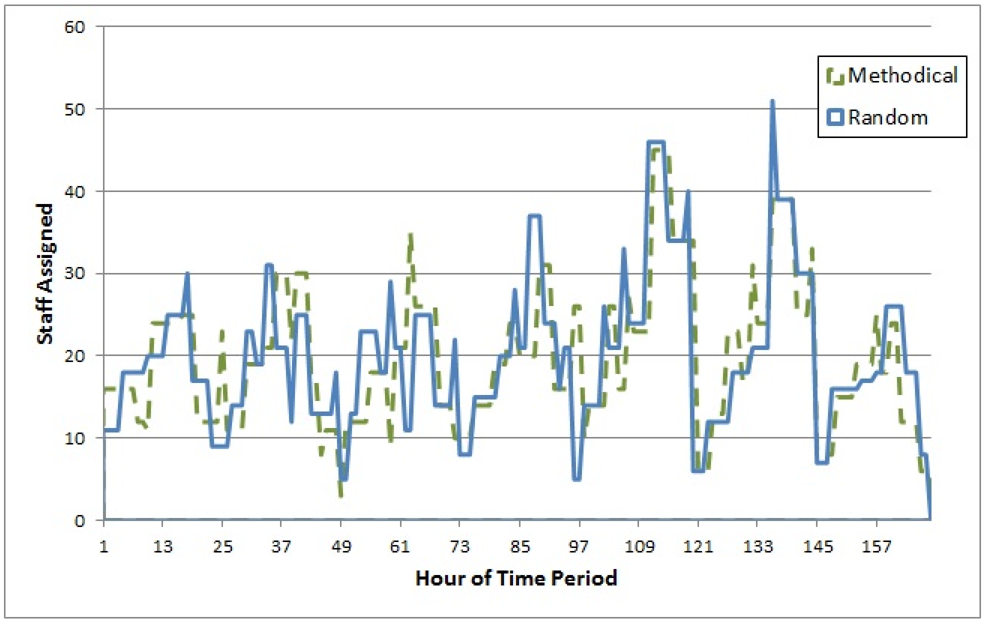


Figure 5.11: Graphical comparison of results from both methodical and random shift length variation in adaptive staffing level search.

5.9.1 Summary

Overall it has been seen that the tabu search process that is best suited to resolution of the target problem makes use of the following features which have been discussed within this chapter:

- Multiple minima diversification in ensuring that other local optima within a close neighbourhood of the first found are investigated.
- Restart diversification to allow perturbation of shift starting times of the initial solution, encouraging a wider search of the potential solution space.
- Restart intensification in providing a stronger starting solution for the main tabu search to initiate from.

- Inclusion of adaptive total staffing level with methodical shift length variation to find the minimal staff across the most optimally defined shifts in order to meet all expected demand.

5.10 Conclusions

Through application to this problem it has been seen that restart diversification is capable of providing a strong improvement to solution quality obtainable by tabu search. Application of the weaker multiple minima diversification and a restart intensification approach were seen to perform less favourably overall. However a less restrictive variant of the restart intensification method did provide an improvement to the rate at which improving solutions were generated. A new objective function was defined that attempted discovery of solutions that used the minimum number of staff members across the best defined shifts in order to meet all of the initially specified demand. Investigation into this approach saw that the best performing method involved iterative increase in total staffing allocation across a defined roster with a methodical method used to manipulate the length of shifts used. Key areas addressed were:

- Introduction to diversification and intensification techniques in order to perturb the shift structure of optimizing rosters.
- Consideration of a new objective function defined through allowing variation in the total staffing level.
- Application of all these techniques to the tabu search methodology presented in the previous chapter.

Through application of these techniques it was seen that:

- The inclusion of restart diversification, multiple minima diversification and restart intensification were all beneficial to the improvement of solution quality and as such would be recommended for use on similar problems.
- Adaptive total staffing provided a method to create solutions whereby all demand is met by a minimum number of staff across a best defined series of shifts.
- Extra freedom in the search gained through variation in shift duration was able to reduce the total additional staff required in adaptive staffing level search.

The following chapter will discuss an alternative optimization technique, that presented by genetic algorithms, and an application of them to the Police staff rostering problem. Genetic algorithms have been widely used within the literature as a comparison against the tabu search method and it forms a natural choice for use as a comparative method within this work.

Chapter 6. Genetic Algorithm Implementation

6.1 Introduction

Though the problem has been successfully tackled using a tailored tabu search optimization algorithm, there are other methods also widely used within the literature upon similar problems. It is of benefit to compare the results obtained through application of at least one of these alternative approaches. This will allow comparison between the results of tabu search and any alternative approach; this may highlight a weakness of the tabu search methodology, be it in calculation cost or in the quality of generated solutions.

Within this chapter the basic methodology behind one of these alternative methods, genetic algorithms, will be discussed in order to provide understanding of the mechanisms governing their operation. This will review both the method by which the standard iterative search process is conducted and solution perturbation mechanisms that provide additional strength and capability to the search. A commercial software package named modeFRONTIER was investigated for use in implementation of a genetic algorithm for this research and accordingly this software and the results of its use will be detailed. However a weakness is identified within this software that may not be easily overcome and as such a C++ based genetic algorithm that is tailored to the Police staffing problem is presented. This algorithm is tested and is shown to possess the ability to generate solutions of comparable quality to those generated by the tabu search method; importantly however it is noted that the genetic algorithm is capable of providing such solutions with a lower calculation time.

6.2 Genetic Algorithms

Genetic algorithms provide a unique and innovative method that may be tailored to suit a variety of optimization tasks. They are inspired by evolutionary biology and make use of techniques based around the ideas underpinning modern theory on evolution. Indeed the first cases in which genetic algorithms were employed was by scientists seeking to model natural evolution using early computers in the late 50s and early 60s, however it was not until 1975 that genetic algorithms were more rigorously defined [54].

6.2.1 Basic Method

The method of implementing genetic algorithms in order to perform an optimization or search process will begin with creation of a suitably sized population of candidate solutions for the defined problem. Commonly these are expressed as a string of elements used to represent a series of numerical values; it is most often seen that these strings are recorded in binary representation as a series of 1's and 0's. For any examples used within this section it will be assumed that the binary representation of candidate solutions will be used. The way in which this initial population is constructed can vary with application, however it is very usual that they are created randomly through generation of a series of random strings of the correct length.

The original population of candidate solutions can now be considered to be the 'parent' population, the members of which will be used in the creation of a new 'child' population of solutions. Each member of this new generation of solutions will be assessed for their strength as measured by a suitable objective function; the function chosen should provide a gauge of how effective the candidate solution is at solving the desired problem. It is then usual for the child population to be cropped by removal of the weakest members in order to encourage progression towards final system solution or optimization. This cropped population is then set as the new parent population and is used in order to generate a third generation of candidate solutions. The third generation is then cropped and used in creation of a fourth generation and this process repeats similarly until a solution is found suitable to the needs of the problem or until variation within the population becomes highly diminished; this reduction in variation would be suggestive that a local or global optimum solution has been found. The method by which a new generation of candidate solutions is generated from a population of parents is highly customizable. The commonest approaches are to use a combination of 'inheritance', 'mutation' and 'crossover'. More complex adjustments to the process in order to illustrate the adaptability and power of genetic algorithms are also possible.

6.2.2 Inheritance

The first and most widely used method of child population generation is that of inheritance. In this parent strings are usually coupled, either randomly or by following a predetermined scheme, and are then used collectively in order to generate single or multiple child strings. Inheritance is accomplished by the direct transference of string data from parents strings to a child string; so the n^{th} element of the child string will most usually be obtained through consideration of the n^{th} elements of the parents. For example consider that the 2 binary strings

$$10011001 \qquad 01010101$$

are chosen to be a coupling of parents in order to generate a new string. Taking data from both of these strings in order to create a child could happen in a large number of ways; a few examples of resulting children from this pairing and the rules used in order to generate them are included in Figure 6.1.

10010101	Taking directly the first 4 elements of the first parent and the last 4 elements of the second.
00110011	n^{th} child element is 1 if both parents have the same n^{th} element and 0 otherwise
$XX01XX01$	Randomly selecting either of the n^{th} parent elements for the n^{th} element of the child; elements that could take the value of either 0 or 1 are denoted using X.

Figure 6.1: Sample results of genetic algorithm child generation using inheritance.

6.2.3 Mutation

Mutation is a method by which additional random variation and diversity may be introduced into a genetic algorithm procedure; this is often used in addition to a child generating procedure such as that given by inheritance. In early genetic algorithm research this was the only operator used, it was much later that the notion of inheritance was proposed [13]. It is implemented by selecting a number of elements within a potential solution string, either randomly or by choice, and then assigning each of them with a random or different value. So, for example, taking the binary string 10011001 and applying the mutation operator to the 3^{rd} element would result in either the exact same string or alternatively 10111001. It is also possible to consider enforcement of a mutation operator wherein any selected string element would be forced to be perturbed by setting it to the opposite of its current value; in this case the sample string would be forced to take the value 10111001.

When mutation is applied it is most usually only done so to a small portion of the current proposed solution set; in this way diversification is encouraged whilst still preserving many of the stronger solution strings. If instead it was decided upon to mutate every solution within a population then it would be highly likely that a good deal of the progress made towards an optimizing solution would be lost. Essentially the result of global population mutation would be entirely random; as such in a situation nearing optimization it would be far more likely to damage than improve the solution using this method.

However considering only the mutation of single elements in a limited amount of strings is a positive diversification method when the search is approaching optimization. This will force the search method to continue exploring potentially distant alternative solutions and will thus, over a long search process, be far more likely to result in a global optimum solution as opposed to only a local one.

6.2.4 Crossover

Similarly to mutation, crossover is commonly used in addition to other string generation procedures and is encouraged in order to provide a larger range of potential solutions. The method is simply achieved by selecting 2 elements with the desired string and interchanging them with each other. So considering the string 10011001 once more and electing to implement the crossover procedure on the 3^{rd} and 4^{th} elements would result in the string 10101001.

As with mutation it should be encouraged only to implement this procedure on a small section of the population; again this is to ensure that stronger and more optimal solutions are retained whilst still allowing a suitable depth of exploration within the solution space.

These 3 techniques provide a good basis for a genetic search algorithm which encourages variation and diversity of solutions in order to more thoroughly explore the set of potential solutions. The research of Hu et al. [14] investigates the capabilities of all of these techniques and applies each of them to the problem of airport gate assignation. It is concluded that for the intended task, genetic algorithms form a viable and efficient optimizing search tool.

The real power of genetic algorithms however is the true level of customization available which allows their use to be specifically tailored to individual problems; potentially yielding higher quality solutions and reducing required computational time.

6.2.5 Variations in Fitness Based Cropping

Variation in the basic procedure used in order to crop a parent population before production of a new child population has potential to provide a variety of different benefits to a genetic search algorithm. Commonly in any genetic algorithm only the strongest parent strings are used in order to create the next child generation; relative strength being measured for each solution string by use of some suitable objective function. In this way then strong parents are coupled with the expected result of a created child string having a similar strength; thus child strings will tend to explore close neighbourhoods of the solution space around their respective parents. Essentially then this approach can be classified as one in which the most fit, or strongest, parents are kept for further population at each iteration of the search algorithm.

Alternatively it is plausible to instead consider keeping the least fit parents in any given generation. In doing this greater diversity of generated solutions is encouraged and a much larger area of the solution space available could be explored. This will allow a genetic search algorithm much greater ability to avoid becoming trapped around an attractive local optimum; thus it will tend to explore a series of local optima, where possible, and suggest the best of these as a final result.

An expansion on these approaches would be to seed the child population with the strongest (or weakest) members of the parent generation before cropping to a new parent population. This guarantees that some members of the generating population will either have suitable strength to encourage solution resilience or suitable weakness to encourage search diversity.

6.3 The modeFRONTIER Software

modeFRONTIER is a multi-objective optimization and design environment software with a range of statistical analysis, modelling and optimization tools [55]. For the purposes of this research, interest was solely upon the automated optimization capabilities that it presented and in particular upon the genetic algorithm methods that were available.

6.3.1 modeFRONTIER Implementation

In order to implement the genetic algorithm capability presented within the modeFRONTIER software it was necessary to redefine the staffing problems as previously used for testing of the tabu search algorithm in a suitable manner. One of the key inputs required was a series of potential solutions to the problem that are represented in strings of suitable values. Fortunately modeFRONTIER is able to automatically generate a series of strings with elements that are randomized within some pre-specified bounds. Such a set of bounds that were constructed here allowed for strings to be created with variation in shift starting times of up to 2 hours from those presented by the initial roster used as input for the tabu search method. Further staffing levels were allowed to take any value randomly selected from the range from 0 to 20 for each shift. An example string for this problem would follow a repeating pattern composed of sub strings of the form XXXYYZZZ, where the X elements indicate shift starting time, the Y elements the shift duration and the Z elements the number of staff assigned. Thus a string reading 00709010 would indicate a shift starting at hour 7 that lasts for 9 hours with 10 assigned staff members. Each string was assessed through an objective function the same as that used within the tabu search methodology, that being calculation of the total amount of surplus expected demand over the time period being considered. This would allow more immediate comparison of results between the 2 different techniques. The genetic algorithm however would be expected to vary string elements outside of the bounds of the tabu search, as a strict ruleset is not enforced to allow staffing level variation it is possible for strings to breed in such a way that staffing levels would maximize and lead to the generation of unrealistically high values. To compensate for this the modeFRONTIER software allowed definition of constraints upon constructed strings that are used in order to mark individual strings as valid or invalid. Such a constraint was created based upon the total number of staff members assigned to a roster represented by each string. The total numbers were set for each of the regional problems to be the same as the total number of staff as spread across each of initial rosters used by the tabu search method. For example, the City region problem for tabu search was defined such that 350 staff members were initially assigned across all defined shifts and redistribution of these staff was sought. Thus for the genetic algorithm interpretation of this problem a string validity constraint was created such that if a string were to represent a roster with more than 350 staff then it would be marked as invalid.

A few trial searches were conducted using the genetic algorithm in order to test the validity of application of the method. These searches were conducted using a set of 32 initial strings, 31 of which were generated randomly whilst the final string was representative of the default minimum cover pattern as used for initialization of the tabu search approach. A mutation and crossover rate of 5% were implemented within all of these trial searches. However, results from these tests were confounded by the fact that the minimum cover solution was seen to propagate very strongly throughout future generations of solutions with very little improvement provided to final solution quality. Indeed when compared with randomly generated solutions, the default minimum cover based staffing roster is relatively strong. Due to this it is likely that this solution is being selected very commonly for breeding with other solutions and as such many of its characteristics are being preserved between generations. This activity results in a case whereby most solutions can be seen to share some features with the default minimum cover roster, restricting potential search diversity for those features. In particular it is commonly seen that those shifts to which no staff are initially assigned within the minimum cover roster will be assigned no staff within the vast majority of generated solutions. A sample extract from 9 generated solutions and the preset initialization roster are illustrated within Table 6.1 where it is seen that in the majority of cases, the zero elements have persisted to the final solutions. Each row in Table 6.1 describes a single solution comprised of 10 elements; these elements are in ‘start-staff’ pairs in which the ‘start’ values indicate shift starting times and the ‘staff’ values the number of staff members assigned. Solution 1 describes the initialization solution and solutions 2 through 10 describe solutions generated by the search algorithm. However, through use of tabu search it was seen that distribution of staff across these shifts led to the creation of higher quality solutions.

Table 6.1: Generated solutions and initialization solution from preliminary modeFRONTIER searches.

	Start1	Staff1	Start2	Staff2	Start3	Staff3	Start4	Staff4	Start5	Staff5
Initializing Roster	**7**	**6**	**9**	**0**	**15**	**6**	**18**	**0**	**22**	**6**
Solution 1	7	4	9	0	16	9	17	0	22	5
Solution 2	7	4	11	0	16	9	18	0	22	5
Solution 3	7	3	10	0	16	8	18	1	22	6
Solution 4	7	5	9	0	15	9	18	0	22	4
Solution 5	7	4	10	2	14	8	16	0	22	4
Solution 6	7	4	9	0	15	9	19	0	22	5
Solution 7	7	4	8	0	13	8	18	0	22	6
Solution 8	7	3	10	0	16	9	18	0	22	6
Solution 9	7	4	9	0	16	8	18	0	22	6
Solution 10	7	4	9	0	15	8	17	2	22	5

In order to compensate for this, there are several options available. First amongst these is to remove the minimum cover based solution from the initial population of solution strings. This would result in an effectively unguided starting point for the search which, though less likely to become confounded by inclusion of a strong starting solution, would be expected to take longer to converge to an optimizing result. Table 6.2 describes the best solution discovered by the genetic algorithm applied after varying amounts of generations have been created. The particular values presented in the table are the objective function score associated with the roster represented by the best solution found as assessed by the same objective function applied within tabu search. It is shown that without the minimum cover solution included within the first generation to guide the start of the search that it requires 1000 full generations to be completed in order for solution quality to near that obtained from only 100 generations with the solution included.

The tenfold increase in calculation time required in order to complete 1000 as opposed to 100 generations immediately invalidates this as a practical option for problem optimization. Indeed the ability to obtain a similar quality of solution after use of far less computation would suggest that inclusion of the minimum cover roster within the starting population is

Table 6.2: Quality of best found solutions after varying numbers of generations with and without strong initial solution.

Generations	50	100	150	200	250	500	750	1000
With solution	233.22	174.16	157.89	137.07	130.89	130.89	130.89	118.70
Without solution	435.65	311.01	311.01	287.86	269.32	224.98	191.03	164.15

of great benefit to the power of the algorithm. Figure 6.2 displays a comparison of a single day within the rosters resulting from genetic algorithm search both with and without the known initial population member. Within this day it is seen that the search without the guiding solution implements an evening shift starting at 18:00 whereas the guided search distributes staff only across shifts as defined by the known guiding solution.

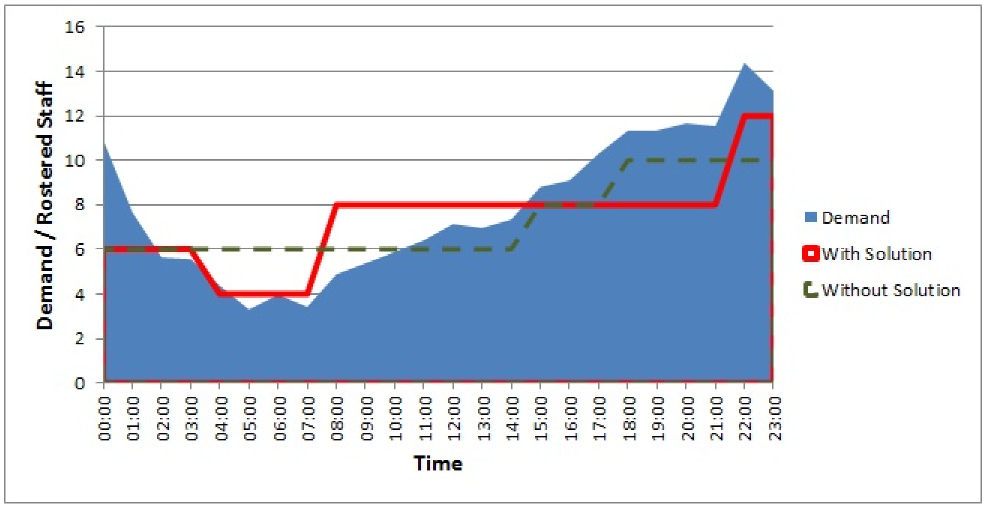


Figure 6.2: Comparison of solutions obtained through genetic algorithm with and without guiding initial population string.

With removal of the offending starting population member being demonstrated as not a suitable option a second alternative approach was considered that relates to the testing performed in consideration of the first option. This second method was through consideration of increasing the number of generations that are created through use of the genetic algorithm in order to assess the rate of convergence of solution quality. Though the values contained within Table 6.2 provide an overall view of the change in solution quality as the generation limit is increased, it would be more useful to create a higher resolution image of the results. To this end Figure 6.3 displays a graphical representation of all those strings amongst the first 3000 generated that provide an improvement to the solution quality represented by the minimum cover solution.

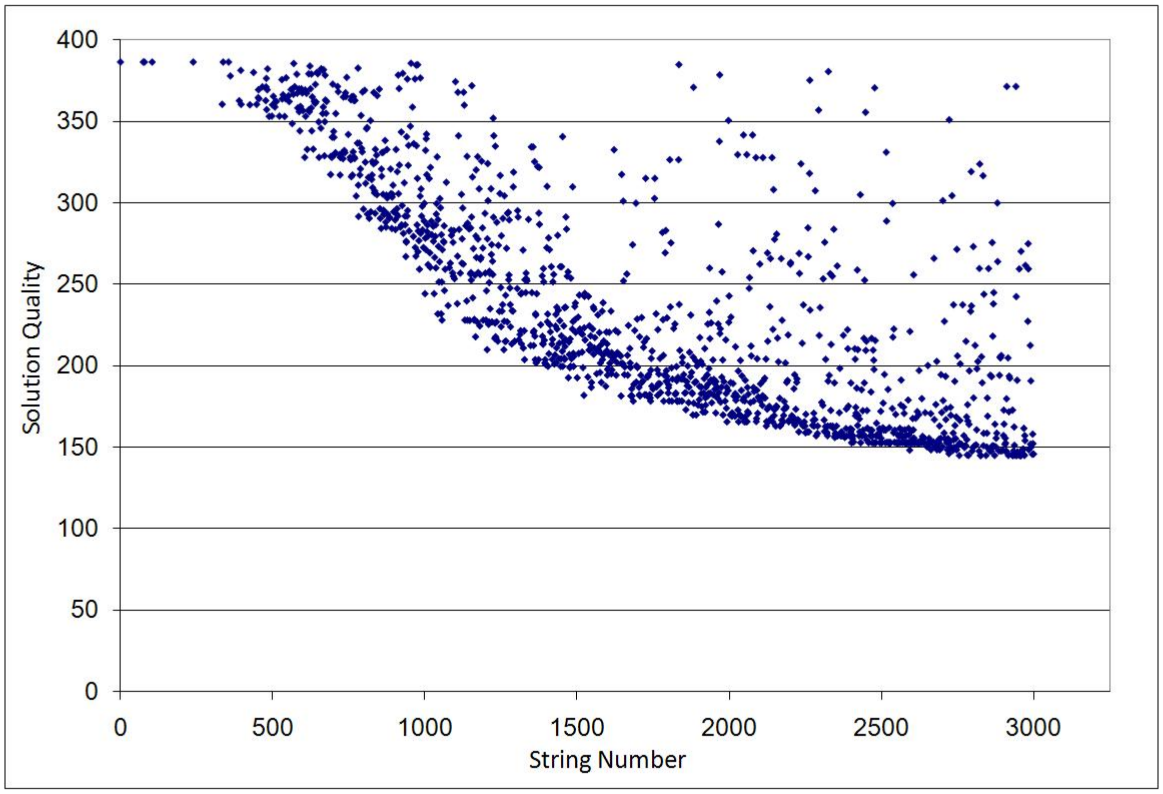


Figure 6.3: Improvements in solution quality as genetic algorithm progresses.

Though the data collected and displayed in Figure 6.3 gives a good representation of the change in rate of best found solution quality improvement there is the possibility that the linear style behaviour emerging towards the end of the graph will not continue. In order to investigate this, the next 3000 generated solutions within the same search were considered and again those that improved upon the initial best solution were represented graphically. Figure 6.4 contains the extended dataset spanning those solutions offering improvement from within the first 6000 solutions generated. This indicates that the apparent linear improvement to best found solution quality continues beyond the first 3000 solutions. Unfortunately this linear nature does not continue indefinitely and solution quality is not seen to improve beyond an objective function value of 100 within a reasonable number of generations. With the high calculation time (multiple hours) required in order to reach such solution quality it may be stated that such an optimization approach would not be suited for live practical implementation within the UK Police industry.

There is still potential that this method and software could be suitable for use in planning

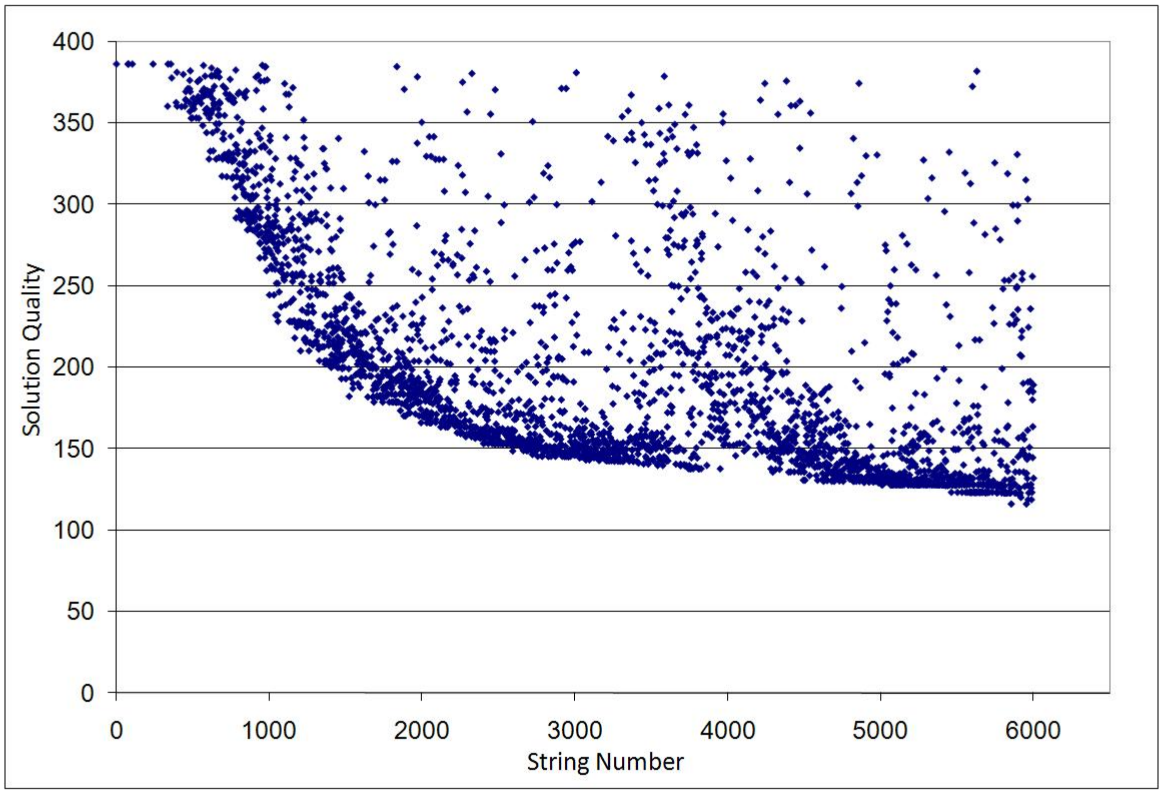


Figure 6.4: Improvements in solution quality as genetic algorithm progresses over an extended period.

required staffing commitments far in advance, and as such the method is not wholly invalid as it still offers an approach towards solution of the target problem. Due to this, further investigation into other variable factors within the genetic algorithm was required in order to find the maximum effectiveness with which it may operate. The remaining major factor suitable for variation that could be expected to provide notable effect upon solution quality was the mutation rate. This mutation rate is the chance that any particular child string will be selected for mutation during the child generation process. The mutation acts upon a set percentage of the elements within a selected string and randomly assigns new values to each of them. A series of mutation rates were selected for use that consider a range of values up to what could be reasonably expected to yield positive results; with a mutation rate too high, the search would essentially be too randomized and the effectiveness of the genetic algorithm reduced. The number of generations for each of these trial searches was set to 50 and the initial guiding string representative of the currently used shift roster was included. Within 50 generations some amount of progress towards optimization should be anticipated and comparison of results obtained after this amount of generations an indication of the speed on convergence will be provided. The population of each generation of strings was again set to 32 and a crossover rate of 5% was used. The results of this series of trials are displayed within Table 6.3 where it is shown that the best solution quality is seen with a mutation rate of 7.5%. Indeed it would be expected that a low mutation rate would be ineffective in diversifying the generated solution strings sufficiently to impact solution quality greatly whereas a high mutation rate should decrease solution quality due to high randomization. This is exactly as demonstrated within the results shown in Table 6.3.

Table 6.3: Change in solution quality provided by modeFRONTIER GA as mutation rate varies.

Mutation Rate	2.5%	5%	7.5%	10%	12.5%	15%	17.5%	20%
Trial 1	253.78	232.68	242.37	236.35	250.54	294.99	207.30	304.02
Trial 2	240.02	220.38	258.59	191.24	282.36	285.51	265.25	256.01
Trial 3	245.89	261.54	187.00	308.24	250.72	288.42	279.13	247.84
Trial 4	259.57	254.16	215.93	297.88	282.95	245.71	227.81	266.21
Average	249.82	242.19	225.97	258.43	266.64	278.66	244.8725	268.52

The mutation rate offering fastest progress towards optimization has been identified. Using searches with lower numbers of generations in order to discover the best performing mutation rate allowed for a reduction in required computation time. Following this a search using this mutation rate with an extended number of generations was conducted; this was in order to attempt discovery of solutions of a higher quality than those previously illustrated in Table 6.2. Other criteria for the search were set to be the same as those previously used, that is with a population limit of 32 strings per generation and a crossover rate of 5%. The results from this search are shown in Table 6.4. Though it is seen that convergence occurs faster using this mutation rate, there is no strong improvement to the solution quality of the best result found by the search.

Table 6.4: Quality of best found solutions after varying numbers of generations with best mutation rate.

Generations	50	100	150	200	250	500	750	1000
Solution Quality	224.84	164.33	149.82	131.01	122.51	122.51	119.25	118.12

Another factor available for variation within this genetic algorithm implementation is the size of the population of each generation of strings, indeed it would be expected that a larger population should lead to greater search diversity. To test this a series of trials were conducted upon the same trial problem; for these trials the best found mutation rate of 7.5% was used with a crossover rate of 5% and generation limit of 200 solutions. The results displayed in Table 6.5 summarise this work by displaying the quality of best found solutions after every 50 generations for each population limit. These resutls indicate that indeed, lower population limits are seen to reduce the optimization ability of the search by not allowing as much variation in the potential pool of string elements at each iteration of the search. Of note, the same number of strings will have been generated after 200 generations with a population limit of 8 and with 50 generations with 32 population limit. For purely the number of solutions considereed to be an important factor these results should be of similar value, however this is not the case and indeed the increased diversity provided with the increased population limit of 32 is shown to provide higher quality solutions after the same total amount of string generation. Such a trend does not continue with higher population limits and it is seen that doubling the limit to 64 provides only very slight increases to solution quality despite the search considering twice the number of strings. Thus it may be stated that there is a point at which the benefit to be gained from the diversity created by an increased population limit would be negated by the associated increase in required calculation cost. Indeed within this research it would appear that within the range of reasonable quality solutions that the most improvement to solution quality per generated solution string is afforded through use of a popultion limit of 32 instead of 64.

Table 6.5: Quality of best found solutions with variation in population limit per generation.

Generations	50	100	150	200
Population 8	351.12	315.70	280.39	261.12
Population 16	301.66	288.51	261.94	231.84
Population 32	224.84	164.33	149.82	131.01
Population 64	210.26	159.65	144.47	129.94

The overall best quality solution found using this search algorithm was provided using 32 strings per generation over 1000 generations with a mutation rate of 7.5% and a crossover rate of 5%. The objective function score associated with this best solution was 118.12. This result is rather weak when compared against the solution quality afforded by tabu search in which solutions for the same trial problem were commonly seen to obtain scores around 24 when assessed by the same objective function; the best solution found having a score of 23.4397. Tanle 6.6 compares an excerpt of the rosters associated with the best found solutions using each of the optimizing search approaches. A table is presented in which the columns describe key elements for each of the rosters. The 'Tabu start' and 'Tabu staff' columns describe shift starting times and staff assigned for shifts in the solution provided by tabu search respectively and similarly the 'GA start' and 'GA staff' columns describe the same information for the solution resulting from use of genetic algorithms. From this it is seen that the genetic algorithm approach implemented retains shifts to which no staff are assigned whereas the tabu search approach is seen to monopolize on the redistribution of staff across all avbailable shifts.

Table 6.6: Comparison of best solutions found using tabu search and modeFRONTIER genetic algorithm.

Tabu start	Tabu staff	GA start	GA staff
7	7	7	129
7	10	014	13
13	1316	11	16
022	11	22	1131
13	31	1333	1
33	0		

6.4 C++ Based Approach

With a weakness apparent with application of the modeFRONTIER genetic algorithm method to this problem it was necessary to explore alternative methods by which such an algorithm could be used. In order to achieve this a customized genetic algorithm was created using C++ as part of this research that was tailored towards solution of the Police resource allocation problem. A major benefit provided through following this process was the ability to more immediately and clearly identify precisely how the search algorithm was acting and to allow custom definition of mutations specifically designed to suit this problem.

6.4.1 The Search Algorithm

The overall process by which the C++ genetic algorithm is performed is described by the flow chart contained within Figure 6.5. The first stage requires the input or automated generation of suitable data from which the problem may be initialized, this involves the input of data indicative of both the supply of and demand upon resources. Secondly, each of the input initial potential solutions (represented by strings or arrays of values) have to be evaluated through consideration of a suitable objective function. This allows each potential solution to be assigned a selection rate for breeding which relates directly to its relative strength and using these rates parent strings may then be paired off in order to allow for the breeding stage to occur. Once a child generation has been created through breeding and mutation, the search process will check to see whether a suitable number of generations have been conducted. If so then the search will terminate and if not then the child generation will become a new parent generation and will undergo the same quality assessment and breeding procedure as used for the prior parent generation.

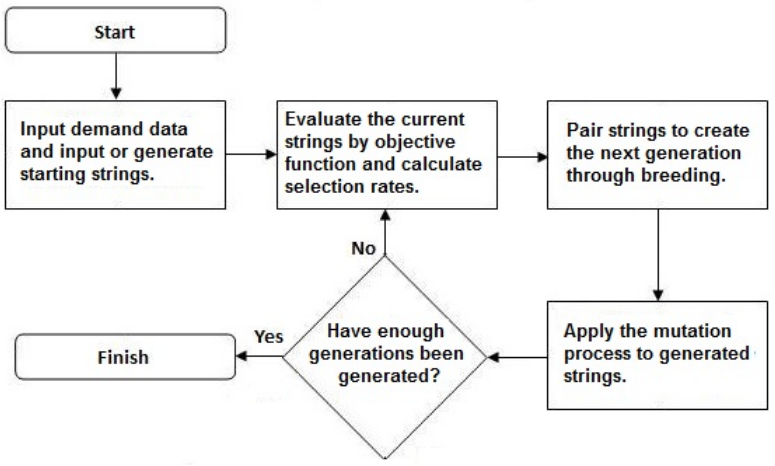
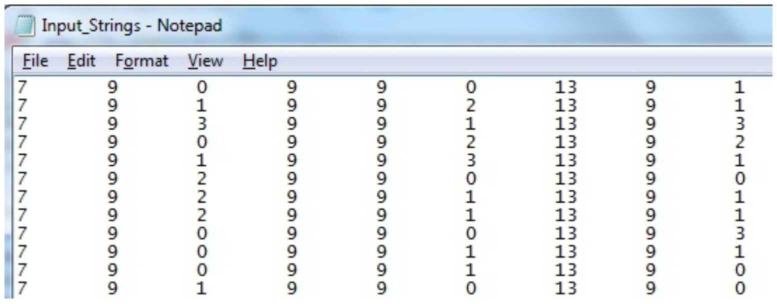

Start
Input demand data and input or generate starting strings.
Evaluate the current strings by objective function and calculate selection rates.
Pair strings to create the next generation through breeding.
Apply the mutation process to generated strings.
Have enough generations been generated?
No
Yes
Finish

Figure 6.5: Flow chart describing the genetic algorithm created using C++.

6.4.1.1 Data Input and String Generation There are 2 primary types of data that need to be input or otherwise generated in initialization of the genetic algorithm. These are an array of data values representative of the expected demand level and a series of strings that are used in representation of potential staffing rosters. Demand data are input from an external .txt file that contains values indicative of hourly expected staffing requirement levels covering the entire time period of interest, these values having been estimated using the incident profiling software described within Section 3.9. These demand data values are stored within an array which is named 'Demand[i]' where the i^{th} element of the array will contain the expected demand level for hour i.

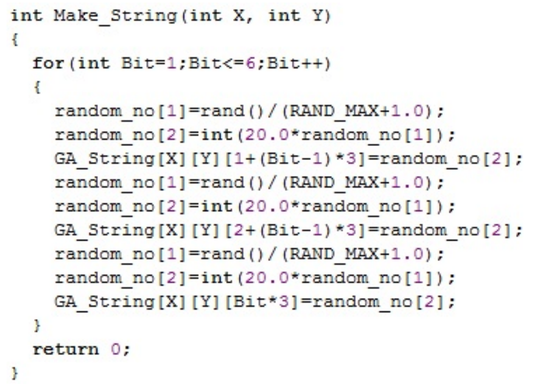
There are 2 options present for the initialization of the strings that the genetic algorithm will use for the optimization process. The first of these method is to read a series of pre-constructed input strings from an additional data file. This is achieved again through the use of a .txt format data file which contains, in each row, an entire string with individual elements tabulated in columns; an example of such a data file which was used in this research is presented in Figure 6.6. The strings are constructed such that each sequential group of 3 elements will describe all necessary features of an individual shift, with the first element describing shift starting time, the second element being the shift duration and the third element detailing initial staffing allocation. For example, in the first string described within Figure 6.6 the first shift would start at hour 7, be of 9 hours in length and be initially assigned 0 staff. Of note, these data values are not represented as a string of 1 and 0 values. The full data file contains 100 strings each with a total of 126 elements (this number being the amount of elements required to describe 6 shifts per day across a 7 day period). A set of 100 strings were created in order to allow subsets of this to be selected as initial populations for any conducted searches; for example, 32 strings could be selected to provide the same population limit as used in the modeFRONTIER testing.



```
7       9       0       9       9       0       13      9       1
7       9       1       9       9       2       13      9       1
7       9       3       9       9       1       13      9       3
7       9       0       9       9       2       13      9       2
7       9       1       9       9       3       13      9       1
7       9       2       9       9       0       13      9       0
7       9       2       9       9       1       13      9       1
7       9       2       9       9       1       13      9       1
7       9       0       9       9       0       13      9       3
7       9       0       9       9       1       13      9       1
7       9       0       9       9       1       13      9       0
7       9       1       9       9       0       13      9       0
```

Figure 6.6: Sample .txt data file extract used for input of pre-constructed strings into the genetic algorithm.

The second approach is to automatically generate an initial series of randomized strings within the C++ program itself. Figure 6.7 provides a sample of C++ code that was used to achieve this. A loop is constructed that performs a string element creation method a number of times equal to the amount of shifts that are desired to be defined; in this code, the loop runs 6 times, thus creating string elements able to fully define 6 shifts. Within this loop 3 elements are individually defined which are indicative of a shift starting time, shift duration and initial staffing level respective of the order in which they are created. The result is an array named 'GA_String[X][Y]' where X indicates the generation (which for the initialization generation will be equal to 1) and Y is a numerical identifier of the shift by order of creation (so the first string will have $Y = 1$, the second $Y = 2$, etc.). This 'Make_String' routine would be called a number of times equal to the total number of strings that are desired to be generated with the input value Y increasing with each successive string. The approach chosen for use within this research was to use a pre-constructed series of strings each time the search was conducted; with investigation into variation of system parameters it was appropriate to start from the same initialization point each time in order to allow more immediate and meaningful comparison of results. It is of note that these strings are not comprised of binary values, with each element instead taking an integer value indicative of a shift defining value. The reason for this is due to the fact that each string defines an entire staffing roster with a variety of shift starting times, durations and staffing allocations. The same string in binary form would be composed of sub elements which would, by necessity, be required to be all of the same length; for example, each shift starting time would need to be represented by a binary value containing the same amount of digits. Thus with shift starting times across a fairly broad range the result would be strings containing many zero values which could propagate strongly throughout the search and confound results. Electing instead to allow integer value representations of each shift defining string element negates the possibility of this eventuality.



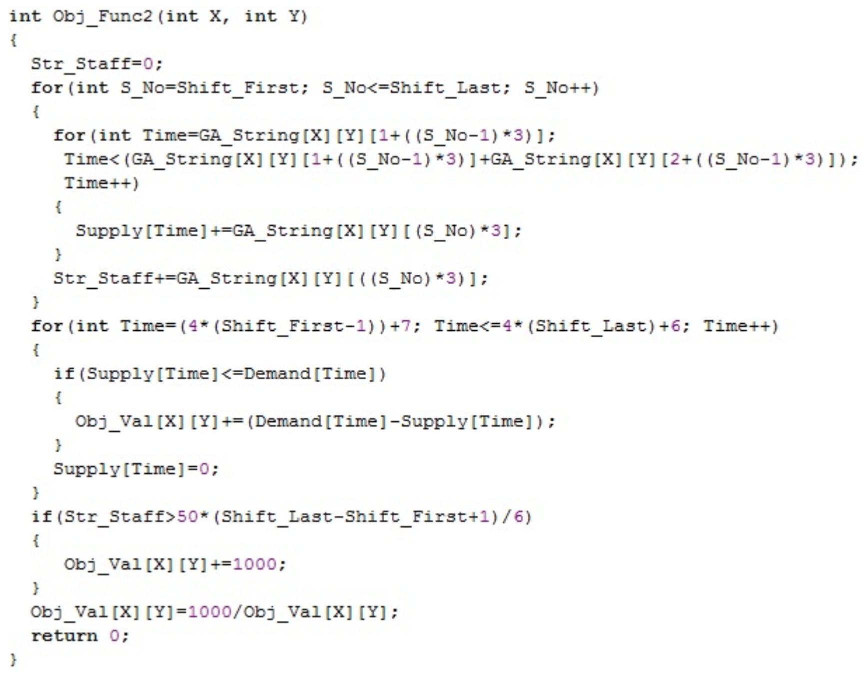
```
int Make_String(int X, int Y)
{
  for(int Bit=1;Bit<=6;Bit++)
  {
    random_no[1]=rand()/(RAND_MAX+1.0);
    random_no[2]=int(20.0*random_no[1]);
    GA_String[X][Y][1+(Bit-1)*3]=random_no[2];
    random_no[1]=rand()/(RAND_MAX+1.0);
    random_no[2]=int(20.0*random_no[1]);
    GA_String[X][Y][2+(Bit-1)*3]=random_no[2];
    random_no[1]=rand()/(RAND_MAX+1.0);
    random_no[2]=int(20.0*random_no[1]);
    GA_String[X][Y][Bit*3]=random_no[2];
  }
  return 0;
}
```

Figure 6.7: C++ code used to generate initial strings within the C++ genetic algorithm.

6.4.1.2 Objective Function and Selection Rate An important factor with implementation of a genetic algorithm is the inclusion of a suitable objective function by which the relative strength of each potential solution string may be quantified. The suitable objective function for use is the same as that described within Chapter 4 for use with the tabu search algorithm, namely that the quality of any potential solution be the total amount of expected demand surplus to that met by the staffing distribution it represents. This alone is not sufficient for application within a genetic algorithm however as the total staffing allocation that a string would represent will be seen to be vary due to the breeding process. Indeed with no penalization to compensate for this it would be expected that the stronger solutions would be those with greater total staffing numbers and as such solutions would be expected to tend towards maximization of these staffing numbers, leading to solutions with unrealistically large values. Figure 6.8 describes a C++ routine used as part of this research entitled ‘Obj_Func2’ that contains such an objective function with a penalty for strings with total staff greater than some defined boundary value. Firstly this code considers each shift defined within the investigated string individually and compiles a time dependent array 'Supply[i]' where the i^{th} value of this array indicates the total staffing availability at time i. To achieve this each point in the 'Supply[i]' array that lies between the starting time (GA_String[X][Y][$1+((S\_No-1)*3)$]) and ending time (GA_String[X][Y][$1+((S\_No-1)*3)$] + GA_String[X][Y][$2+((S\_No-1)*3)$]) for every shift is selected and increased by the amount of staff associated with that shift (GA_String[X][Y][$((S_{N}o)*3)$]). Next for each point over the entire time period considered the routine then checks whether the amount of expected demand is greater than the available supply of staff, in the case that it is an objective value 'Obj_Val[X][Y]' is increased by an amount equal to the surplus demand. From this a resultant 'Obj_Val[X][Y]' value is obtained that is indicative of the total surplus demand that would be expected for string number Y of generation X. This is then further supplemented through the inclusion of a penalty that acts based upon the total amount of staff and number of shifts present within the roster described by the solution string. A check is made to see if the amount of staff is greater than 50 per calendar day considered; with the problem defined to allow 6 available shifts per day this is given by $50*(\text{Shift\_Last}-\text{Shift\_First})/6$. If a string represents a roster in breach of this condition then a penalty value (in this case 1000) is applied to the objective function value associated with it in order to discourage its selection at the breeding stage of the genetic algorithm. This penalty ensures that the desired amount of staff members are allocated across the roster and that the search does not just maximize the amount of staff assigned unrealistically.

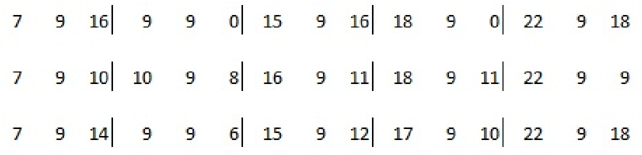
As an example, consider the 3 strings presented in Figure 6.9. Each of these is a string that represents the shifts defined within a single calendar day with 5 groups of 3 elements each; these groups representing shift starting time, shift duration and staff allocation in order of element appearance. The first of these strings is indicative of the assignation of 50 total staff ($16+0+16+0+18$), this string would not be penalized based upon this day. The second string uses only 49 staff ($10+8+11+11+9$) and as such it too would not be penalized. The final string however makes use of 60 staff in total ($14+6+12+10+18$) and so the objective function score associated with this string would be heavily penalized in order to indicate the unrealistic staffing level and to discourage selection of it in the breeding process.

These objective values are used in order to create a series of selection rates (or probabilities) representative of the chance that each string will be selected as a parent for generation of a child string within the breeding process. However as the goal is minimization of surplus demand it is necessary to further adjust the objective values to make them suitable for this purpose. Stronger solutions are those with a lower amount of surplus demand, and thus a



```
int Obj_Func2(int X, int Y)
{
  Str_Staff=0;
  for(int S_No=Shift_First; S_No<=Shift_Last; S_No++)
  {
    for(int Time=GA_String[X][Y][1+((S_No-1)*3)];
     Time<(GA_String[X][Y][1+((S_No-1)*3)]+GA_String[X][Y][2+((S_No-1)*3)]);
     Time++)
    {
      Supply[Time]+=GA_String[X][Y][(S_No)*3];
    }
    Str_Staff+=GA_String[X][Y][((S_No)*3)];
  }
  for(int Time=(4*(Shift_First-1))+7; Time<=4*(Shift_Last)+6; Time++)
  {
    if(Supply[Time]<=Demand[Time])
    {
      Obj_Val[X][Y]+=(Demand[Time]-Supply[Time]);
    }
    Supply[Time]=0;
  }
  if(Str_Staff>50*(Shift_Last-Shift_First+1)/6)
  {
     Obj_Val[X][Y]+=1000;
  }
  Obj_Val[X][Y]=1000/Obj_Val[X][Y];
  return 0;
}
```

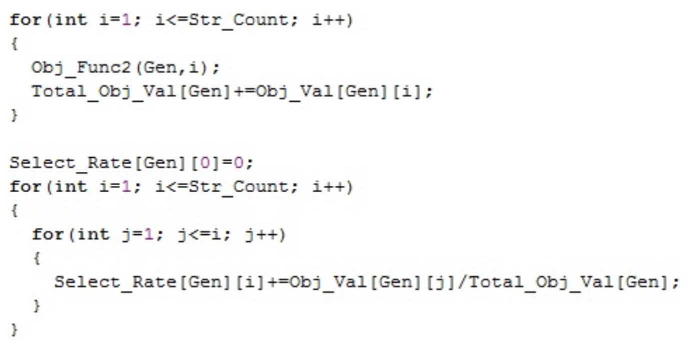
Figure 6.8: C++ code used as objective function to evaluate each string considered using the genetic algorithm.



7	9	16	9	9	0	15	9	16	18	9	0	22	9	18
7	9	10	10	9	8	16	9	11	18	9	11	22	9	9
7	9	14	9	9	6	15	9	12	17	9	10	22	9	18

Figure 6.9: Example strings from C++ based genetic algorithm implementation.

lower objective value, in order to obtain larger selection rates for these it is suitable then to calculate the reciprocal objective value for each string. A value of 1000 is used as the numerator so that those strings which have been penalized in breach of the upper total staffing amount will be immediately notable as those with resulting objective function values lower then 1. In order to translate these into selection rates for each of the strings a C++ routine as shown in Figure 6.10 is used. The routine first calculates a cumulative total objective value for all of the current generation strings combined ('Total_Obj_Val[i]' where i represents the generation in question). Using this a selection probability for each individual string is calculated as the score associated with that string divided by the total score for all strings combined. In this way a string with higher associated score will then have a higher selection rate. The value 'RAND_MAX' is equal to the maximum value that a randomly generated integer may take, thus division of a randomly generated value by this maximum will create a random real value between 0 and 1.

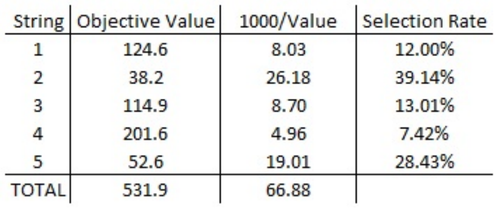


```
for(int i=1; i<=Str_Count; i++)
{
  Obj_Func2(Gen,i);
  Total_Obj_Val[Gen]+=Obj_Val[Gen][i];
}

Select_Rate[Gen][0]=0;
for(int i=1; i<=Str_Count; i++)
{
  for(int j=1; j<=i; j++)
  {
    Select_Rate[Gen][i]+=Obj_Val[Gen][j]/Total_Obj_Val[Gen];
  }
}
```

Figure 6.10: C++ code used to assign each parent string with a selection rate.

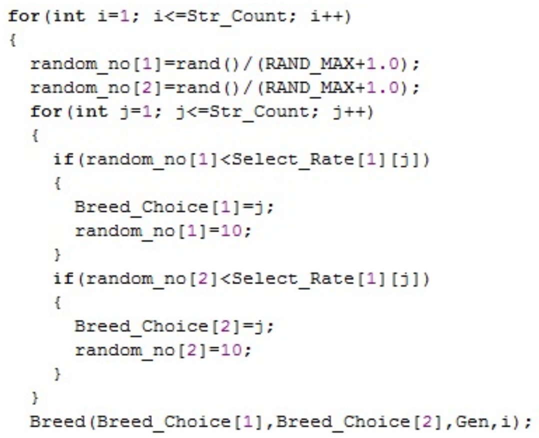
Important to note is that the selection rates are compiled in such a way that each subsequent rate is the sum of itself and all previously calculated rates. For example, if the first selection rate were calculated as 12% and the second at 8% then this would be recorded by setting the first selection rate value to 12% and the second to 20%; in this way the final selection rate will always be 100%. The reason for this adjustment is to allow the generation of a single random number between 0 and 1 to be sufficient in identifying a string for selection, regardless of the total objective function value of all strings. Table 6.7 contains a further example of this with sample objective function values for a population of 5 strings. Considering the first of these strings, an objective value of 124.6 is translated through reciprocation and mutliplication by 1000 to the value 8.03. The selection rate is generated from this by consideration of the sum of all such values for all strings in the current population, in this case the sum is 66.88 resulting in a selection rate of 12% for string 1.

Table 6.7: Sample objective function value and string selection rates used in C++ genetic algorithm.



String	Objective Value	1000/Value	Selection Rate
1	124.6	8.03	12.00%
2	38.2	26.18	39.14%
3	114.9	8.70	13.01%
4	201.6	4.96	7.42%
5	52.6	19.01	28.43%
TOTAL	531.9	66.88	

6.4.1.3 Breeding the Next Generation of Strings In order to generate new child strings it is necessary for suitable pairing of the parent generation to be formed. In order to achieve this a routine is used that makes use of the calculated selection probabilities in randomly selecting pairs of parent strings, code for which is displayed in Figure 6.11. With the particular method by which the selection rates are compiled, generation of 2 random values between 0 and 1 here is suitable for identifying 2 shifts to be used as a pair. The random values generated are compared alongside the selection rate values for each possible parent string until the correct string is identified, at this point the identified string is recorded and the generated random value is set to an unfindable value in order to effectively terminate the search.

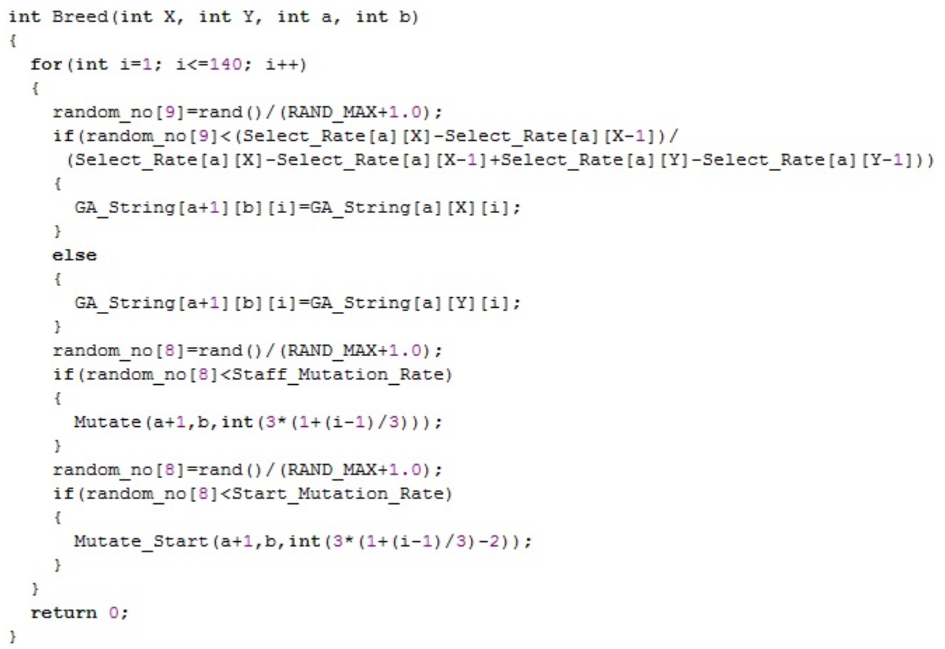


```
for(int i=1; i<=Str_Count; i++)
{
  random_no[1]=rand()/(RAND_MAX+1.0);
  random_no[2]=rand()/(RAND_MAX+1.0);
  for(int j=1; j<=Str_Count; j++)
  {
    if(random_no[1]<Select_Rate[1][j])
    {
      Breed_Choice[1]=j;
      random_no[1]=10;
    }
    if(random_no[2]<Select_Rate[1][j])
    {
      Breed_Choice[2]=j;
      random_no[2]=10;
    }
  }
  Breed(Breed_Choice[1],Breed_Choice[2],Gen,i);
```

Figure 6.11: The process by which parents are selected for breeding in the C++ method.

Once a pair of suitable strings have been found the string selection process will then call a separate routine entitled 'Breed()' which is used to perform the breeding operation using the 2 selected strings. 'Breed(Breed_Choice[1],Breed_choice[2],Gen,i)' will cause the breeding operation to occur upon the 2 selected parent strings in creation of a new child string of number i that will belong to the next generation of strings (Gen+1). It is possible that the same parent may be selected twice through this process however this is suitable as it allows the possibility that some strings from the parent generation may be carried through to the child generation without an additional fitness based cropping stage being required in the search algorithm. Of note, this is more likely to occur for stronger strings, which are those most likely to have strong beneficial characteristics that would be desired to keep in future generations.

The breeding process itself occurs using the code that is demonstrated within Figure 6.12. The pair of parent strings are considered alongside each other, with an element in one of them being paired with its relative element within the other; in this way the first elements of both parents are considered, then the second elements of both, and so on. For each element pairing the relative strengths of each parent are considered when deciding which element should be passed on to the child string. This is accomplished by considering the total selection rate of both parent strings and then creating selection rates based upon the strength of each parent as part of this total selection rate. Thus for example, the pairing of 2 strings with selection rates of 12% and 8% would result in element selection rates in their breeding of 60% (12/20) and 40% (8/20) respectively. This results in the situation that stronger parent strings are more likely to pass on their characteristics in the breeding process in addition to being more likely to be selected for it. When this process has been conducted for each element of the parent strings then a new string will have been created as a semi-random combination of them and will have been identified as a member of the child generation. Considering again the string selection rate details in Figure 6.7, each pairing of 2 strings here will result in different element selection rates during the breeding process. For example the pairing of strings 1 and 2 would yield a sum of their individual selection rates of 51.14% $(12\% + 39.14\%)$, this then leads to an element selection probability of 23.46% for string 1 (12% 51.14%) and 76.54% for string 2 (39.14% 51.14%).



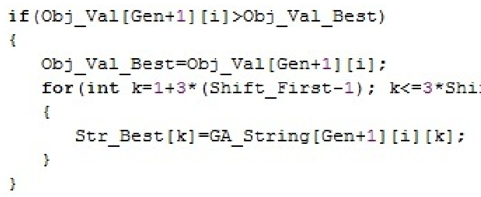
```
int Breed(int X, int Y, int a, int b)
{
  for(int i=1; i<=140; i++)
  {
    random_no[9]=rand()/(RAND_MAX+1.0);
    if(random_no[9]<(Select_Rate[a][X]-Select_Rate[a][X-1])/
     (Select_Rate[a][X]-Select_Rate[a][X-1]+Select_Rate[a][Y]-Select_Rate[a][Y-1]))
    {
      GA_String[a+1][b][i]=GA_String[a][X][i];
    }
    else
    {
      GA_String[a+1][b][i]=GA_String[a][Y][i];
    }
    random_no[8]=rand()/(RAND_MAX+1.0);
    if(random_no[8]<Staff_Mutation_Rate)
    {
      Mutate(a+1,b,int(3*(1+(i-1)/3)));
    }
    random_no[8]=rand()/(RAND_MAX+1.0);
    if(random_no[8]<Start_Mutation_Rate)
    {
      Mutate_Start(a+1,b,int(3*(1+(i-1)/3)-2));
    }
  }
  return 0;
}
```

Figure 6.12: C++ code describing the parent string breeding and child string generation process.

At this stage of the process, mutation routines that perturb associated shift starting times and staffing levels may be called for each of the newly defined elements of the child string.

These calls are achieved through random probabilistic chance based upon mutation rates that are defined and stored within floats titled 'Staff_Mutation_Rate' and 'Start_Mutation_Rate' for the staff level perturbation and start time perturbation respectively.

As the strength of each string is quantified through the objective function analysis process it is possible that one result will be stronger than the current best. In this case the program automatically updates a record of the best found string, replacing the older record with the newly discovered stronger solution. This is achieved through the code shown in Figure 6.13 and results in an array 'Str_Best[i]' that records the elements of the strongest string found as the search progresses. As the search terminates after a specified number of generations, this array will then dictate the most optimal solution that was found throughout the entire search process; thus effectively providing the solution to the problem.



```
if(Obj_Val[Gen+1][i]>Obj_Val_Best)
{
   Obj_Val_Best=Obj_Val[Gen+1][i];
   for(int k=1+3*(Shift_First-1); k<=3*Shift_Last; k++)
   {
      Str_Best[k]=GA_String[Gen+1][i][k];
   }
}
```

Figure 6.13: Method by which the GA written in C++ updates best found string records.

6.4.1.4 Mutation In order to allow solution variation beyond the values contained within the original 100 defined strings it was necessary to implement a mutation mechanism within the search which would allow perturbation of each individual element to occur. Due to the variety of elements represented within each of the solution strings 2 different mutations were required, the first allowing perturbation of shift staffing allocations and the second perturbing shift starting times. C++ code that was used to implement staffing level element perturbation is demonstrated within Figure 6.14. This code generates a value that may be positive or negative in order to perturb the currently used staffing level, either by increasing or decreasing it respectively. Importantly the range of values that this perturbation may take are integer values between -1 and $+2$, in this way it would be seen over repeated applications of the mutation that the total staffing level across all strings would increase. Coupling this with the already described solution quality penalization for a string with too many staff members, the best final solutions found will be those that make the best use of precisely this limiting number of staff. Thus coupling the penalization method with this type of staff increasing mutation it is possible to specify precisely the target number of staff desired to be seen within a final generated roster.

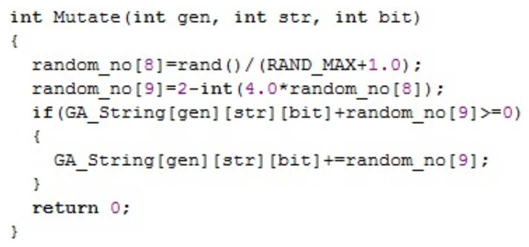

```
int Mutate(int gen, int str, int bit)
{
  random_no[8]=rand()/(RAND_MAX+1.0);
  random_no[9]=2-int(4.0*random_no[8]);
  if(GA_String[gen][str][bit]+random_no[9]>=0)
  {
    GA_String[gen][str][bit]+=random_no[9];
  }
  return 0;
}
```

Figure 6.14: C++ code through which staff level string elements are perturbed by mutation.

The second mutation perturbation acting instead upon shift starting times performs similarly in that it adjusts the element to which it is applied by either increasing or decreasing it. There is no strong benefit to encouraging global increase or decrease of all shift starting times over the course of a search process however and so equal probabilistic weighting is placed upon both possibilities. Thus the chance that a shift will have a starting time perturbation increasing the start time by 1 hour is equal to the chance it will be decreased by 1 hour. Code that was used in order to achieve this mutation is shown in Figure 6.15.

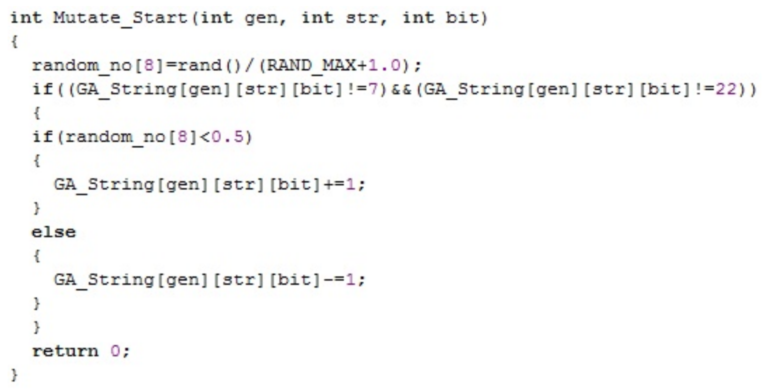

```
int Mutate_Start(int gen, int str, int bit)
{
  random_no[8]=rand()/(RAND_MAX+1.0);
  if((GA_String[gen][str][bit]!=7)&&(GA_String[gen][str][bit]!=22))
  {
  if(random_no[8]<0.5)
  {
    GA_String[gen][str][bit]+=1;
  }
  else
  {
    GA_String[gen][str][bit]-=1;
  }
  }
  return 0;
}
```

Figure 6.15: C++ code through which shift starting time elements are perturbed by mutation.

6.4.2 Results of C++ Genetic Algorithm Implementation

In order to first trial the constructed genetic algorithm a series of trials were conducted upon the same first problem considered for the modeFRONTIER software testing with a similar set of system factors. In this way the same population of 100 potential solution strings was input as the search initialization point and the search was allowed to construct 1000 child generations with the best solution found at specific benchmark generations noted. Within this search a mutation rate of 5% was used. The results of this first search are presented in Table 6.8.

Table 6.8: Quality of best found solutions after certain numbers of generations using the C++ based genetic algorithm.

Generations	100	250	500	1000	5000	10000
Best Solution Score	943.63	407.294	283.524	210.249	170.320	153.999

This result proved to be similar to that shown by the modeFRONTIER software in that it would take a very large amount of generations in order to begin to approach solutions of similar final quality to those given by tabu search. Indeed with the reduced computational cost required in order to implement this C++ based approach it was possible to extend the number of constructed generations to 10000 whilst still requiring dramatically less total calculation time than the modeFRONTIER algorithm. For example, the modeFRONTIER algorithm was seen to require approximately 20 minutes per 3000 generated solution strings; this would mean that with a population limit of 30 strings it would take in the region of 20 minutes to create 100 generations. The C++ algorithm however requires approximately 5 seconds in order to create the same number of total strings. However when compared to the results shown in Table 6.2 it is demonstrated that the C++ algorithm converges slower towards a suitable result.

It was demonstrated with the modeFRONTIER algorithm that adjustment to the population limit of each generation may result in adjustment to the quality of the best found final solution. Accordingly it was also appropriate to perform such system variation for the C++ based genetic algorithm in order to ascertain whether an increase in population size will markedly improve solution quality. Indeed it is also possible that a reduction in the initially trialled population limit of 100 strings may provide solutions of similar quality; thus requiring less computational effort in order to produce the same results. A series of searches were conducted using a range of population limits, the results of which are displayed in Table 6.9. Within each of these searches mutation rates of 5% were used and the total number of generations created in each case was 1000. From these results it may be seen that population limits significantly lower than 100 strings result in detriment to solution quality, indeed with a low population limit of 25 objective function value is seen to almsot double from that offered using a limit of 100 strings. Further, increasing the population limit significantly does not result in strong improvement to solution quality and it may thus be stated that a population within the vicinity of 100 is most well suited for this algorithm on the problem considered.

Table 6.9: Quality of best solutions obtained with population limit variation within the C++ based algorithm.

Population Limit	25	50	100	150	200	250
Best Solution Score	410.651	243.820	210.249	209.145	209.231	207.726

An option that was considered in order to compensate for the previously noted slow rate of convergence of the search was in reducing complexity of the problem by performing a separate genetic algorithm search for each day of the week, thus reducing the amount of search string elements dramatically and similarly reducing the number of potential solutions within each search. This was initially conducted within the modeFRONTIER software however in that case no strong benefit was provided and indeed it was seen that results were obtained of a very similar overall strength. This means that when the total objective scores for all 7 individual day problems were totalled it was very similar to the score provided when conducting a problem spanning the entire week at once. In contrast, performing such an adjustment within the C++ algorithm led to a dramatic decrease in the total objective function score as the algorithm was seen to generate far stronger solutions for each individual day which provided a strong overall solution when compiled as a full week. Table 6.10 contains data describing results from conducting both a single week problem and a day by day breakdown based problem using both modeFRONTIER and the C++ algorithm. These problems were conducted using a total of 1000 generations each with a population of 100 strings in each generation with a mutation selection rate of 5% in both cases. The total staffing level constraints in each of the daily problems were adjusted to compensate for the fact that greater demand was expected to occur on Friday and Saturday, however the total number of staff across all 7 days remained constant at 350. In order to apply this methodology then a separate search is performed for each of the 7 calendar days in which the solution strings are segments of the previously defined string containing the elements that are associated with each particular day. From this it may be seen that not only does the C++ algorithm strongly outperform the modeFRONTIER algorithm when the problem is redefined into a series of daily problems, but indeed it is even capable of generating results of a similar quality to those provided by the tabu search methodology.

A graphical representation of the highest quality roster found using compilation of 7 daily rosters is presented in Figure 6.16 with comparison alongside the best found solution using

Table 6.10: Comparison of weekly and daily problem results using C++ and modeFRONTIER.

	Mon	Tue	Wed	Thu	Fri	Sat	Sun	SUM	Week
modeFRONTIER	18.21	21.55	19.03	24.83	26.12	30.92	12.61	153.27	144.96
C++	0.86	2.29	2.36	2.62	8.12	9.26	0.00	25.51	210.25

tabu search. There is strong similarity between the 2 full weekly rosters displayed, with many defined shifts covering the same period of time and indeed receiving the same staffing allocation. The few differences that remain between these rosters coupled with the similarity in solution quality suggest that the solution is reaching global optimality, this being shown by the situation in which no roster adjustments are able to provide more than a very minor improvement to solution quality.

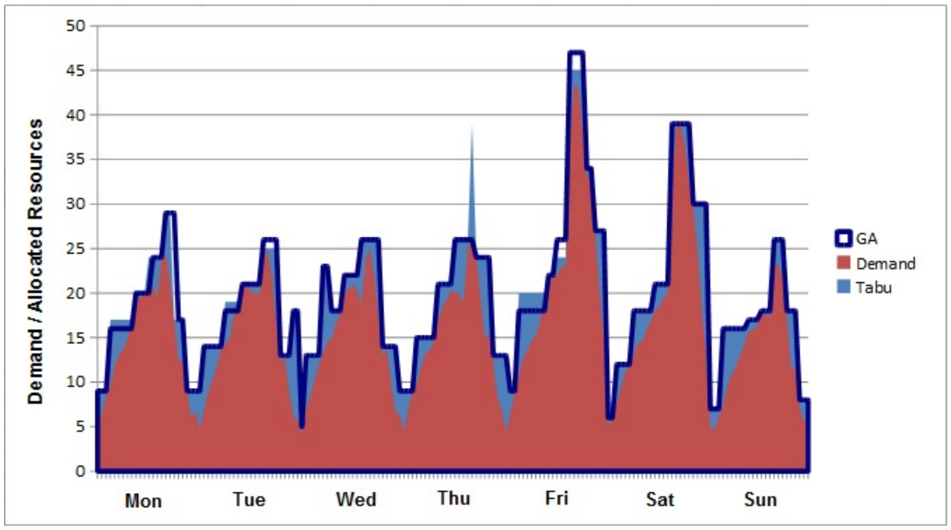
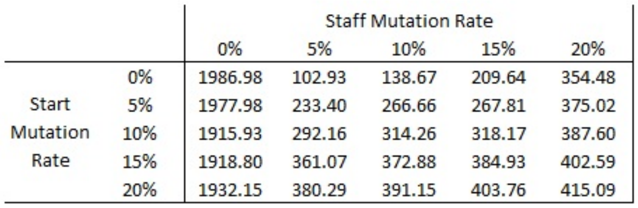


Figure 6.16: Comparison of results from tabu search and daily genetic algorithm application.

With the potential of the C++ algorithm established it was necessary to further investigate potential improvements that could be obtained through the variation of key system parameters, namely the 2 rates of selection for string mutation. Indeed it was demonstrated within Figure 6.3 that a particular range of mutation selection rates were able to provide higher quality solutions in the modeFRONTIER application and thus a similar investigation was carried out upon the C++ algorithm. A range of selection rate values were trialled for both the starting time and staffing level perturbations with test implementation on the same base problem. This problem was conducted using a population of 100 strings per generation with 1000 generations being conducted within each search. The results from these tests are displayed in Table 6.11 where the objective function scores for the best found solutions using each combination of mutation selection rates are shown.

Table 6.11: Results of variation mutation within the C++ based genetic algorithm.



		Staff Mutation Rate				
		0%	5%	10%	15%	20%
	0%	1986.98	102.93	138.67	209.64	354.48
Start	5%	1977.98	233.40	266.66	267.81	375.02
Mutation	10%	1915.93	292.16	314.26	318.17	387.60
Rate	15%	1918.80	361.07	372.88	384.93	402.59
	20%	1932.15	380.29	391.15	403.76	415.09

The most immediate result noticeable from Table 6.11 is that any trial conducted in which there was 0% chance of mutation of staffing level string elements performed extremely poorly with objective function scores of greater than 1900. The reason for this is due to the effect that this staff level mutation has in slowly raising the overall staffing level across all strings as the search progresses. Indeed it was earlier mentioned that the coupling of this mutation with the staffing level based string validity constraint would provide solutions using precisely the target number of staff. Without this mutation, many strings will be generated using far fewer staff than the constraint value and with no impetus to encourage growth in the staffing levels it is seen that the resulting best solutions have not made full use of as many staff as are possible. This underuse of available resources then leads to the situation in which there is a far greater level of surplus demand, hence the higher objective function scores seen. As noted for the modeFRONTIER application, it is also seen in the C++ algorithm case that for larger values of either mutation rate that final best found solution quality is seen to degrade. The best performing application is shown to be with the use of a 5% chance of selection for mutation of staffing level and 0% chance of shift starting time mutation, indeed even the lowest starting time mutation rate was seen to result in greatly increased final solution scores.

With best system criteria defined it was of interest to consider the convergence of the non-segmented search in which a full week roster is considered as opposed to 7 daily rosters. With the C++ based algorithm able to perform a far greater number of generations within a reasonable amount of time it was possible to consider an extended number of generations and find the point at which solution quality improvements cease. Figure 6.17 displays the improvement to solution quality as the search progresses for 5 separate searches with 15000 generations being created in each case. It may be seen from this that all 5 of the searches converge at a similar rate to solutions of a similar quality with no improvement to solution quality present in any case within the last several thousand generations. From this it may be stated that convergence to a solution of quality equal to that generated using tabu search of the segregated genetic algorithm modification would require an excessively large number of generations, which becomes impractical due to the required computation time.

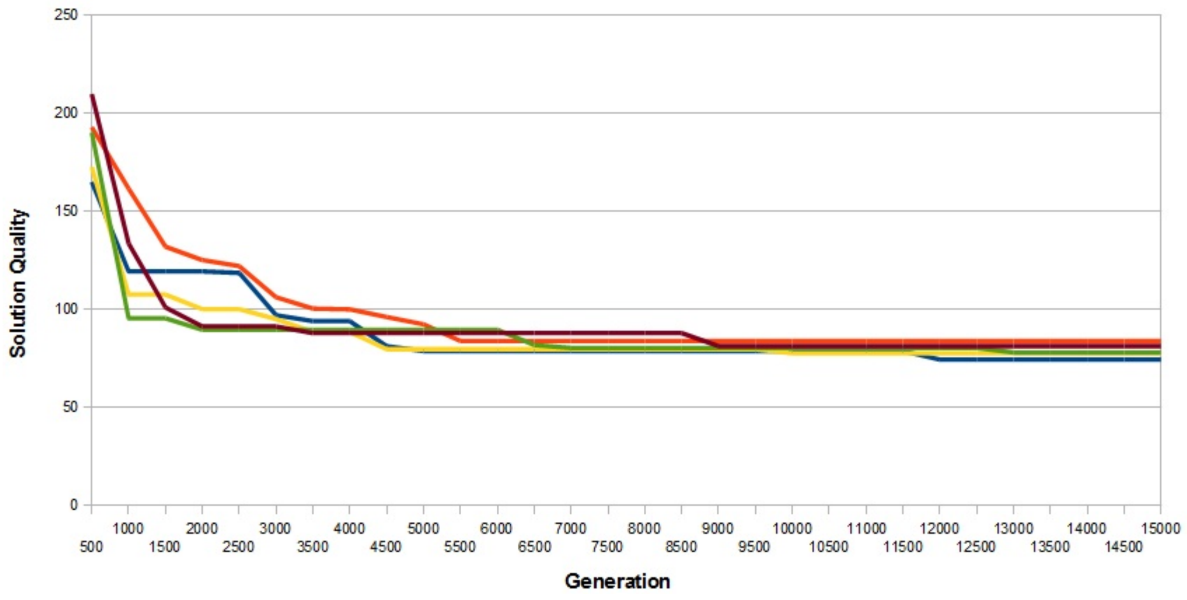


Figure 6.17: Graph displaying convergence of the C++ genetic algorithm.

6.4.3 Summary

From the results of application of the C++ based genetic algorithm it has been shown that the best performing algorithm made use of the following features:

- Redefinition of the base problem into a series of daily problems as opposed to a single weekly one.
- Use of staffing level mutation in order to encourage staffing level growth with a selection rate of 5%.
- A validity constraint imposed upon the algorithm such that any string representative of total staffing above this constraint is heavily penalized.
- No mutation upon the shift starting times throughout the search (effectively achieved through setting selection rate to 0%, negating it).

The C++ based genetic algorithms have been seen to strongly improve upon the performance of the modeFRONTIER application and indeed result in the creation of solutions with a similar quality to those generated by tabu search. This is only achieved through redefinition of the problem into a series of smaller sub-problems however and the C++ genetic algorithm struggles to converge to high quality solutions when applied to a larger problem. The tabu search is seen to outperform genetic algorithms when applied upon the larger problem due to its natural affinity to combinatorial problems of the type considered.

6.5 Conclusions

This chapter has presented an alternative heuristic search algorithm through the use of genetic algorithms and has shown that it is able to create solutions of a similar quality to those generated using the tabu search approach. Initial work using modeFRONTIER identified a weakness in the provided search algorithm resulting in slow convergence to solutions of a reasonable quality and a tendency for weak solutions to propagate throughout later generations, confounding results. The tailored C++ algorithm overcame this problem and through redefinition of the target problem was seen to be a highly successful technique.

One final option that could achieve improvement to the modeFRONTIER method would be to consider a hybridization of the provided genetic algorithms with the tabu search presented within this thesis. This would be achieved by using best found solutions after a reasonable number of generations using genetic algorithms to form the starting solutions for tabu search. This is highly suitable as genetic algorithms provide an intelligent method by which such starting solutions may be defined. Further benefit is obtained through consideration of the identified weakness in the genetic algorithm implemented; particularly that specific shifts are seen to take staffing values of 0 and for these values to persist throughout future generations. The tabu search method implemented has been seen to utilize such shifts and trial distribution of staff across them in order to generate high quality final solutions. Thus application of tabu search to a solution generated by the genetic algorithm would be expected to improve upon solution quality by more thoroughly investigating these shifts. Precisely this hybridization will be discussed and its application detailed within the following chapter.

Chapter 7. Additional Work

7.1 Introduction

Methodologies were considered throughout this research which were set aside in favour of areas that appeared to have more potential. One such methodology was the use of probabilistic methods in creation of demand profiles that are not as rigidly defined through deterministic historical data. At a later stage of the research this proved to be of interest again due to consideration of an alternative objective function that considers target attendance times for incidents of varying grade. Within this chapter work regarding probabilistic demand profile compilation and preliminary work concerning the alternative objective function is shown. Additionally, other areas within the Police organization were considered with preliminary work conducted in order to generate profiles of the demands present upon them. Within this chapter the work concerning a single such alternative department is presented, this being the canine trained unit of Leicestershire Police.

As mentioned in Chapter 6, there is potential to improve upon the genetic algorithm implementation provided by modeFRONTIER by use of a hybridization with tabu search. In this chapter a basic method by which this may be achieved is described and the results of application to the same trial problems as previously considered are discussed.

7.2 Probabilistic Approaches to Demand Compilation

7.2.1 A Simple Method

Through the ideas of probabilistic modelling and inspiration from the field of reliability engineering it is possible to approach the challenge of accurately representing incident based demand in a very different way than that used within the original Police demand profiling tool.

Considering the same basic data as used for the current Police model (namely records of the times and grades for incident calls received by the CMC) this can provide us with a series of values indicating the rate of incident occurrence for individual hours of the day. Manipulating these slightly it is thus possible to obtain a new set of values estimating the amount of time that passes between incident occurrence; these values may be known as Mean Time Between Incident (MTBI) values, given

$$MTBI_i = \frac{60}{R_i}.$$

Where R_i is the incident rate for hour i and i takes values sufficient to cover the desired period of time. The value of 60 is included in order to force the resulting MTBI values to be recorded in minutes; this was chosen in order to allow for more direct comparison between the results provided from the Police demand profiling tool and those obtained probabilistically. It is worth noting that it is possible for any given hour to have an incident occurrence rate (R_i) equal to zero, potentially causing an impossible MTBI calculation. To compensate for this it would be advisable to place some suitable lower bound value on R_i, though the particular value chosen will clearly vary based on the scenario involved and any computational software used.

This notion of MTBI suitable for usage in this application is a direct comparison to that of Mean Time Between Failure from reliability engineering in which the amount of time a system will remain operational is considered. Similarly it is also convenient to adopt the idea of Mean Time To Repair (MTTR) where this value is taken to be the average amount of time that will be taken between officer assignation to an incident and their future release from it. Essentially then a system is constructed in which an incident happening is taken to be a system failure requiring the attendance of an officer over a period of time in order to repair the system back to normality. In usual reliability engineering practices however the system is analyzed with respect to the likelihood of it being functional at any point in time. This clearly is not suitable for the task of demand modelling and thus here the system is never considered to fail and individual failures accumulate over time. This is of particular importance for practical application of the technique to policing practices as it is highly common that the MTTR will be greater than the MTBI.

Both the MTBI and the MTTR values are, as their names suggest, mean values and accordingly it is important to consider that in a real situation there will be considerable variance in actual times both between incidents and to repair. Due to this then a suitable method would be to use probability distributions centred around the provided mean values in order to more accurately model MTBI and MTTR. Fortunately in the case presented by Leicestershire constabulary, sufficient data is available to allow this. For example Table 7.1 describes the mean and variance of incident occurrence rates found for a single policing region within Leicestershire over 2 calendar years. The table presented displays the mean and variance in incident occurrence rates seen across a series of individual hours of a single calendar day. Using such data it is possible to make use of suitable probability distributions; one such example being the Poisson distribution.

Table 7.1: Mean and variance of incident occurrence rates for 8 hours of a sample calendar day.

	07:00		08:00	09:00	10:00	11:00	12:00	13:00
14:00								
Mean	4.45	6.69	8.36	9.84	11.81	13.11	13.69	14.80
Variance	0.76	1.12	1.32	1.24	2.03	2.52	2.18	2.61

Using these ideas a representation of demand over a specified period of time may be generated, however it is important to note that suitable initial conditions will be critical. As this approach is based entirely around the idea that upon the occurrence of one incident it is possible to predict the time of the next it could be possible to reduce physical accuracy by choosing the first incident to happen at a highly improbable time. For example assuming that the initial triggering incident occurs at a time when, historically, very few occur would potentially result in a large overestimation of the demand present at that time. There is also the issue that at the selected initialization there will most likely be many more than 1 incident already underway and that these incidents will each have started at varying prior points in time. Both of these problems may be mitigated by providing some initial conditions and then allowing the system to run from these through a 'warm-up' period first before the desired system initialization point is reached. At a minimum it would be suggested to allow for at least one full calendar day for the warm-up period as this will allow for the natural daily build-up and decline of demand to undergo a full cycle. Issues arising due to selection of an improbably placed initial incident will be resolved more than that caused by an unrealistic amount of initial incidents by doing this. Accordingly it would be expected to minimize inaccuracy by selecting a time of day at which fewer incidents occur to be that at which the initial conditions are set. Historically for the type of demand that is desired to be modelled this time of day is in the region of 6am however precise amounts of incidents that have been seen at this time will vary widely across different regions of the County.

Given then that a suitable set of initial conditions for the system have been found, the model may now begin to run. The model is set to assume upon initialization that an incident has just occurred, the total amount of incidents currently in progress is logged along with the current time. This log is further supplemented by the future time at which the incident will be considered to have ended; this is calculated using the MTTR which can either be taken as a fixed value or allowed to vary in order to encourage more variability in resulting demand profiles. Additionally it is at this point that the start time of the next incident is decided upon by considering the MTBI value for the hour in which the currently starting incident is located. Simulated time then progresses until either the start time of a new incident is reached or the end time of an incident underway occurs. In the case of a new incident start time the same procedure as previously described occurs; the start time and current amount of incidents is logged, the end time for the new incident is generated and a start time for an additional incident is created. Upon reaching a time marked as the end of an incident the total amount of incidents underway is decreased by 1 and the new running total is logged alongside the current time; this process may be summarized by the flow chart shown in Figure 7.1.

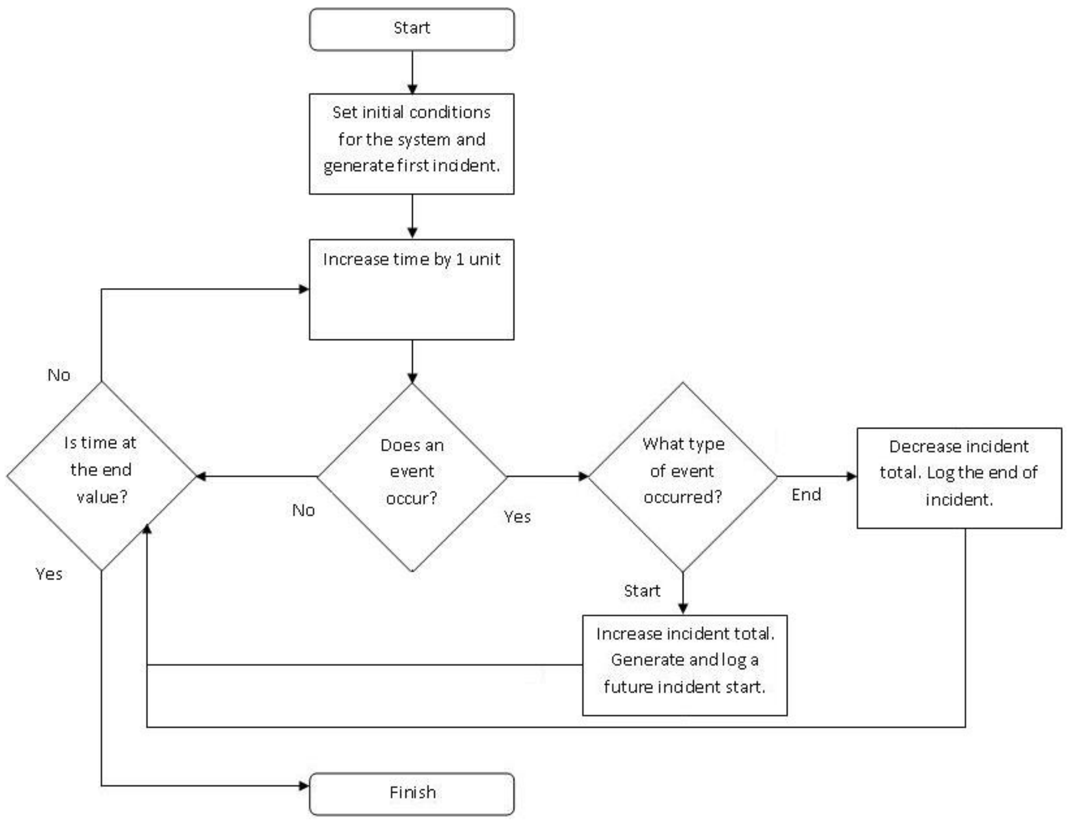

Start
Set initial conditions for the system and generate first incident.
Increase time by 1 unit
No
Is time at the end value?
Does an event occur?
No
Yes
What type of event occurred?
End
Decrease incident total. Log the end of incident.
Yes
Start
Increase incident total. Generate and log a future incident start.
Finish

Figure 7.1: A flow chart illustrating the proposed probabilistic modelling method.

Following this procedure alone will provide a suitable base demand profile, however there is one change which may result in a higher level of realism though this will be in addition to an increase in predicted demand. Consider that it is quite plausible for there to be on average fewer than 1 incidents per hour, resulting in a predicted MTBI for that hour of more than 60 minutes; this raises the possibility to ignore individual hours of the day for use in incident creation. Importantly this would make it possible for a high incident rate hour of the day to effectively be skipped if preceded by an hour in which historically very few incidents have occurred. It is possible to counter this issue by re-evaluating the predicted next incident occurrence time as the boundary between hours is crossed. In the potentially problematic situation described previously, in which a low incident rate hour is followed by a high incident one, it would be of benefit to consider reduction of the time until next incident initialization. To do this it would be suggested to compare the current amount of time remaining until the new incident begins with the MTBI for the new hour being encountered; then in the case that the MTBI is some significant amount lower than the time remaining it would be of benefit to regenerate the time until incident initialization using the new MTBI. This addition to the probabilistic method will thus ensure that more historically incident dense periods are not unintentionally skipped in the generation of a demand profile.

Using the full probabilistic method, including the further consideration of hourly MTBI comparison, it is now possible to generate a realistic representation of the base demand on which further model complexities may later be built. A comparison of demand profiles created using the initial Police profiling tool and the proposed probabilistic method is shown in Figure 7.2; both profiles are generated using the same incident occurrence rates and simulate base demand for the Loughborough region of Leicestershire for an average Tuesday.

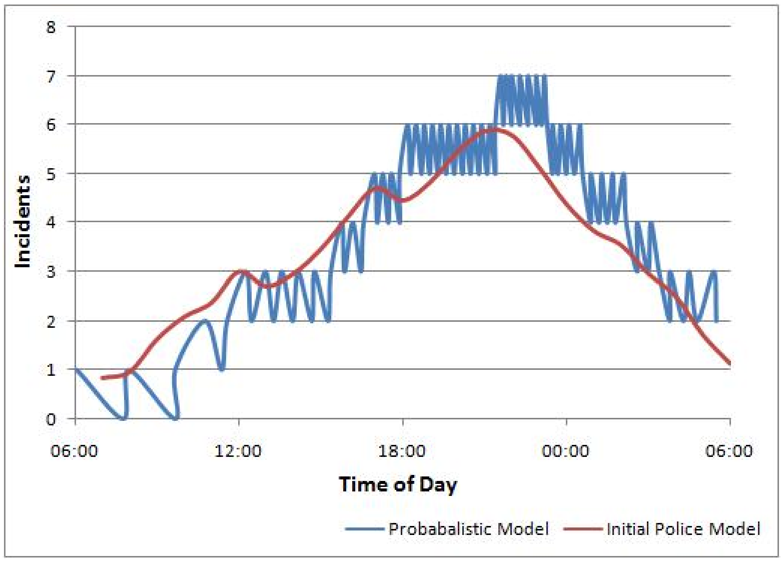


Figure 7.2: A comparison of base demand profiles created with the initial profiling tool and probabilistic method.

There are immediately clear similarities between the illustrated demand profiles, both experiencing the same general increases or decreases in demand levels and sharing the time placement of their daily minimum and maximum amounts. Due to the nature of profile construction, the probabilistic method provides a representation that often rapidly jumps between several integer values; whereas the demand profiling tool uses raw historical averages that are unlikely to take precise integer values. This difference is clearly made visible in Figure 7.2, where smoothly ascending and descending slopes describe demand variation for the profiling tool and sharp jumps are commonly seen in the probabilistic method result. With no data variability and using fixed mean incident occurrence rates the demand predicted through the probabilistic method would be directly comparable to that provided through the deterministic model; with regular oscillation above and below the deterministic profile. A much clearer representation of predicted demand levels from the probabilistic method may be obtained by creating upper and lower bound curves, then stating demand to oscillate between these boundaries. This would result from a simple adaptation of the type of data used to show the probabilistic method in Figure 7.2; simply recording each individual peak as part of the upper bound curve and each trough as part of the lower bound. Doing precisely this for the same series of data (namely that for a Tuesday in the Loughborough region) provides the paired curves shown in Figure 7.3. For example, the maximum live incidents seen between 12:00 and 15:00 is 3 and the minimum over the same period is 2.

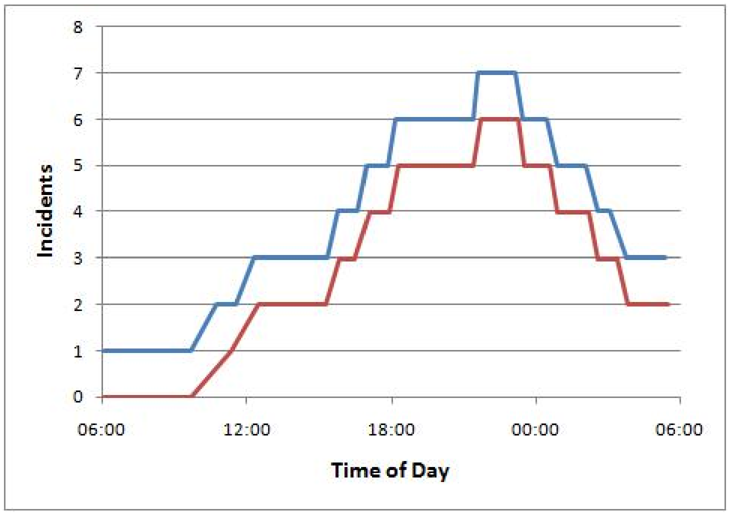


Figure 7.3: Sample upper and lower bounds on demand using the probabilistic method.

This method provides the basis for another approach to the modelling of incident based demand on local Police officer time, and indeed has several advantages over the approach taken for construction of the initial Police demand profiling tool. Primarily the greatest advantage would be the expected gain in accuracy due to the resulting demand profile being updated on a minute by minute basis; whereas the Police profiling tool works through use of hourly or, after the proposed re-segregation, quarter-hourly demand blocks. This will also allow the incorporation of more complex additional demand forms as the base profile is augmented to take account of other important factors; indeed a demand spike of any duration should be simple to include using a model using this base.

Unfortunately there are also drawbacks to be considered upon implementation of this method. As discussed the resulting model could well be anticipated to be more accurate and provide a better representation of the demand in question, however use of the method would almost certainly require a programming platform such as C++ or Python. This in itself would not cause problems for creation of a functional tool, it would however raise the issue of future resilience with the need for continuous development by well trained personnel. The Microsoft Excel based original demand profiling tool is far better suited in this respect as Police staff are far more commonly trained in its usage than they are in any programming language at all, let alone the one that is chosen for creation of a probabilistic modelling tool. The use of such a powerful programming platform does however bring additional benefits; most importantly their usage generally provides a much more effective calculation engine than that given by Microsoft Excel, with a much broader range of calculation options and a massively higher data input limit. The probabilistic results illustrated within Figures 7.2 and 7.3 within this report were calculated using a simple demonstration program written using the C++ programming language.

7.2.2 A More Detailed Approach

In order to provide a more complex probabilistic view of the demand from incidents generated from calls by the general public to the Police Call Management Centre, an alternative approach was taken. The core of this approach lies in defining a series of incidents to cover the period of time of interest; each of these incidents being defined through the generation of several key pieces of information.

The first stage in the process is to identify the volumes of incidents that will need to be generated throughout the time period and at which specific points in time each of these incidents will occur. To do this the historical average rates of incident occurrence per interval within the time period are considered alongside the noted standard deviation of incident rates per interval. For each time interval then these data are used in order to create an estimated number of incidents by use of the formula

$$I(t) = \bar{I}(t) + \sigma(t)(0.5 - R).$$

Where $\bar{I}(t)$ is the average number of incidents for time interval t, $\sigma(t)$ is the standard deviation of incidents for time interval t and R is a random real value between 0 and 1. This formula will then generate an estimate to the amount of incidents occurring within each period by first taking the known average level of incidents for that period and then adding to this some multiple of the standard deviation for the same period; this multiple varying between -0.5 and 0.5. In order to create a realistic amount of incidents for later use it is necessary to round the resulting $I(t)$ values to the nearest natural values. Upon completion of this for every interval within the time period of interest a volume of incidents will have been defined that are distributed amongst these intervals.

These volumes are then used in order to define a series of incidents and the starting times associated with them; this information will form the base of the incident demand profile. This is accomplished by considering each of the time intervals and creating an amount of incidents that will start within them equal to the previously calculated estimated volume amounts. The start times for these incidents are calculated through entirely random distribution over the time interval in which they are located; for example an interval covering a time period from 4:30pm to 4:44pm would allow the incidents generated within it to start at any point between these 2 times.

The only further information required in order to provide an image of the incident based demand is the duration of each of these incidents, however there are many stages required in order to provide an accurate duration the first few of which will define other incident characteristics.

The first such defining characteristic is the location of the incident, that is which of the Leicestershire policing regions it occurs within. Though it was earlier noted as not having a large effect upon the time required in order to resolve an incident upon its assignation to an officer there is still some impact caused by this. More importantly though this will bear an effect upon the other defined characteristics, most notably the likelihood of arrest as this will vary greatly for specific times in particular regions; for example more urban regions are likely to have a higher arrest rate on Friday and Saturday nights than rural areas. The region of each incident is assigned through use of the historical average occurrences for each of the regions in creating a set of probabilities; these describing the probability that an occurring incident will be in each region. These probabilities will sum to 1 as when an incident is known to occur it is impossible that it does so outside of the predefined regions. Thus with creation of a random real value between 0 and 1 for each incident it is possible to assign them each a region. These probabilities will vary however based upon the time of day and as such it is necessary to have a set of such values for each hour of the day in order to create suitably realistic defined incidents. Table 7.2 details a sample of such probability values. In total there are 144 values required in order to allow for suitable probabilistic distribution of incidents across the 6 City LPU regions (24 values for each region).

Table 7.2: Incident region of occurrence probabilities for City LPUs.

Region	CB	CH	CK	CM	CN	CW
Probability 07:00	0.15	0.21	0.16	0.19	0.13	0.16
Probability 08:00	0.16	0.21	0.13	0.23	0.13	0.15
Probability 09:00	0.17	0.21	0.11	0.28	0.11	0.13
Probability 10:00	0.13	0.17	0.12	0.39	0.09	0.10

The next 2 characteristics more fully define the incidents, describing their grades as understood by the Leicestershire incident grading policy and also the category of incident. There are 5 such categories and these are namely anti-social behaviour, crime, public safety, road related and other. For both of these characteristics a similar set of probabilities as used in defining incident locations are used; they are however broken down to be dependent upon different factors. First of these characteristics to be defined is the incident grade with probabilities for this being based upon the incident location; for each region then a set of probabilities that sum to 1 are made to define incident grade. Secondly the incident type is defined based upon both the incident location and grade; for each region and grade pairing, a series of probabilities are made to define the incident type. Important to note is that a set of these probabilities will be necessary for each of the time intervals throughout the period of interest in order to ensure that such variation as the time of day may cause is not ignored. With 3 separate incident grades and 5 incident types, this results in a total of 15 potential categorizations for each generated incident. Each of these separate categories will require different probabilistic chances if occurrence for each incident, which will further require the region and time of occurrence to be considered. With 144 time and region pairings this results in a total of 2160 required incident probability values, a single value for each possible combination of time, region, incident type and incident grade. It is impractical to list so many values within this thesis and indeed with 4 parameters used for definition of each there is no simple way by which they may be tabulated for representation.

With all of these characteristics defined the durations for each incident are now generated using a large series of historical average values. So for every combination of incident region, grade and type and also for every time interval within the period of interest, a value is provided for the average resolution time. This figure is then subjected to a slight random modification through scaling by a multiplicative factor that could take any real value from 0.9 to 1.1; duration will then vary by up to 10% from the calculated average values. This is in order to allow some level of variability to the duration of generated incidents, as would be seen within a real world situation.

As would be expected, every incident that is attended by Police staff has the potential to result in the arrest of one or more people. The likelihood of arrest will vary based upon the type of incident attended, the grade of incident attended and even the time of day or day of the week. Due to this every incident created in the previous stage is considered and through definition of another set of probabilities is either marked as an incident which results in arrest or one which does not. For each time interval, every grouping of incident type, grade and region is assigned a value indicating the probability that the particular combination for an incident will result in an arrest. These probabilities are calculated for each of these groupings by considering the historical numbers of such incidents that have and haven't resulted in arrest; so the probability would be the amount of incidents of that type leading to arrest divided by the total number of incidents of the same type. For each incident that is marked as one that results in arrest, an additional 7 hours is added to the incident duration to simulate the amount of time spent dealing with the custodial process.

At this point a series of incidents have been defined to a sufficient degree as to be able to provide a representation of the incident based demand upon LPO staff. In order to do this the starting time and duration of each incident is considered to represent a block of time for which 1 officer is required to attend that incident. Thus the consideration of all incidents would result in the compilation of many such blocks into a single profile showing the volume of demand over time. As an example Figure 7.4 displays the results of this incident generation process when applied using data for the City Basic Command Unit (BCU) of Leicestershire on a Sunday.

This result behaves as would be expected by the conjectural view of demand within the Police organization. There is however a critical area that should be noted and that is the presence of a certain amount of lead in time wherein demand will be underestimated. This is due to the creation and consideration of incidents only within the period of interest; those occurring before the period in question will also likely bear an important effect upon the

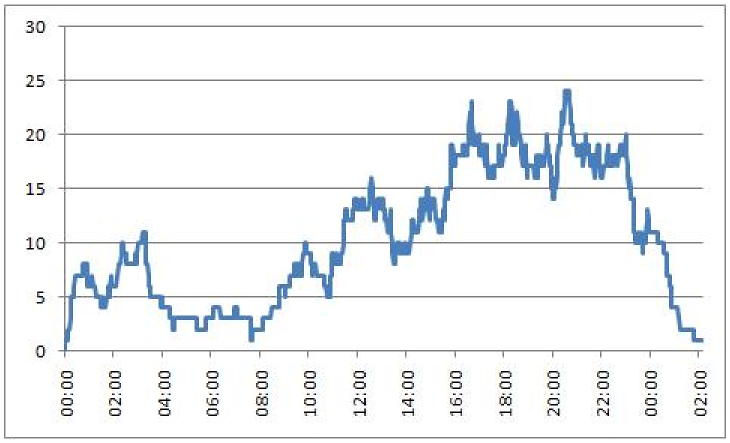


Figure 7.4: Demand profile created for the City BCU of Leicestershire using enhanced probabilistic approach.

demand. To counter this there are 2 major options, the first of which is to state that at initialization there are a certain number of incidents already in progress; these may be defined by use of historical average occurrence rates. Alternatively setting the simulation to run through the 12 hours prior to the period of interest first would allow any incidents, including those resulting in arrest, that would bear an effect to also be simulated. Note again that this methodology provides only an approach by which the demands present upon Police staff may be simulated and does not in itself generate staffing levels and shift definitions. It would be necessary to apply an optimization methodology using the created demand profile as part of the problem definition; for example, this could be in replacement of the demand data presented in the trial problems of Appendix A. A full version of the C++ code used in implementation of this probabilistic methodology is included within Appendix B.

7.3 An Objective Function based on Attendance Target Success Rate Applied with Tabu Search

In addition to the function based upon the matching of volumes of expected demand and staff assigned within a roster, a secondary objective function was created to instead consider the ability to respond to individual incidents within a suitable time frame. Incidents that are graded either as a grade 1 or grade 2 are recognized as those that require either an immediate or rapid response from Police personnel. Accordingly there exist target times within which it is required for a Police Officer to attend these grades of incidents. The target attendance time for a grade 1 incident within Leicestershire is 15 minutes and an objective is set such that 85% of such incidents are to be attended within this time. Similarly grade 2 incidents have a target attendance time of 60 minutes with the objective being to meet 80% of these incidents within this time.

An objective function may be defined based upon these targets by considering the response time required in order to meet individual incidents and calculating the percentage of them that are met within the desired times. Consider a situation in which there exist a number of grade 1 and 2 incidents each of which requires the attendance of Police staff for some varying amount of time. As each incident occurs one of the available staff members will be allocated to it and will remain so until the incident is resolved. In the case that a new incident starts and there are no staff available to attend the incident, it would be placed onto a queue and the time of occurrence marked. This process is summarized by the flow chart presented in Figure 7.5. Should sufficient time pass in which no staff members are available to attend a queued incident within the target time (15 minutes for grade 1 and 60 minutes for grade 2) then the incident would be marked as having failed to be met, yet it would still require resolution through Police attendance and would remain in the queue. In this way then, a more optimal roster would be one that has a distribution of staff suited to allow the most rapid response to newly developing incidents; this being marked with a lower amount of failures. A success rate presented by the roster could then be represented by the number of successes divided by the total number of incidents, this is the percentage of incidents that were met within the target times. Indeed it would be suitable to have a pair of objective functions in order to consider the different target response times and desired success rates.

7.3.1 C++ Implementation of the Attendance Target Objective

The first stage in allowing the application of the newly defined objective function is in creation of a series of simulated incidents by which the suitability of a prospective roster may be assessed. To achieve this a method similar to that presented in Section 7.2.2 was used. In order to generate a detailed series of incidents with variance in the time each requires for resolution it is necessary first to generate volumes of incidents occurring across the time periods of interest and to distribute these appropriately. In order to inform this basic incident volume generation a series of values are input into a C++ program from an external .txt based data file source. Code that is used to achieve this is presented within Figure 7.6 where for every hour within a target time period a series of data values are input. These values include the minimum (‘Incs_Mins[i]’) and maximum (‘Incs_Max[i]’) number of incidents historically seen within that hour of the particular day in addition to the mean average (‘Incs_Mean[i]’) and standard deviation (‘Incs_SD[i]’) in number of incidents seen

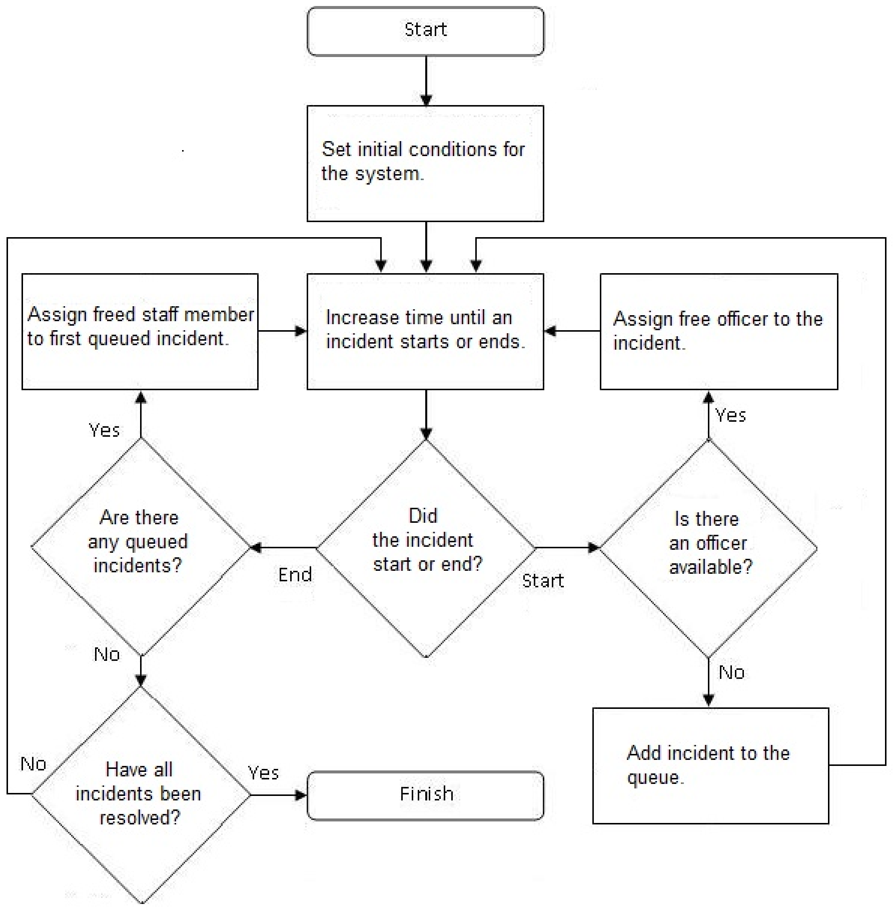

Start
Set initial conditions for the system.
Assign freed staff member to first queued incident.
Increase time until an incident starts or ends.
Assign free officer to the incident.
Yes
Yes
Are there any queued incidents?
End
Did the incident start or end?
Start
Is there an officer available?
No
No
No
Have all incidents been resolved?
Yes
Finish
Add incident to the queue.

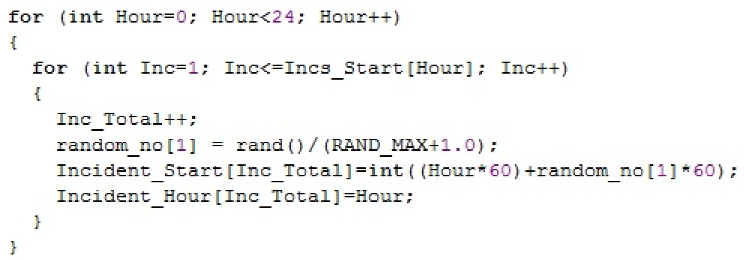
Figure 7.5: Flow chart describing incident attendance and queuing process.

(for each hour i).

```
indata >> fl_DataRead;
while ( !indata.eof() )
   {
   Incs_Min[Step_Count]=fl_DataRead;
   cout<<Incs_Min[Step_Count]<<"\t";
   indata >> fl_DataRead;
   Incs_Max[Step_Count]=fl_DataRead;
   cout<<Incs_Max[Step_Count]<<"\t";
   indata >> fl_DataRead;
   Incs_Mean[Step_Count]=fl_DataRead;
   cout<<Incs_Mean[Step_Count]<<"\t";
   indata >> fl_DataRead;
   Incs_SD[Step_Count]=fl_DataRead;
   cout<<Incs_SD[Step_Count]<<"\n";
   Step_Count++;
   indata >> fl_DataRead;
   }
indata.close();
```

Figure 7.6: C++ code for data input used for incident generation process.

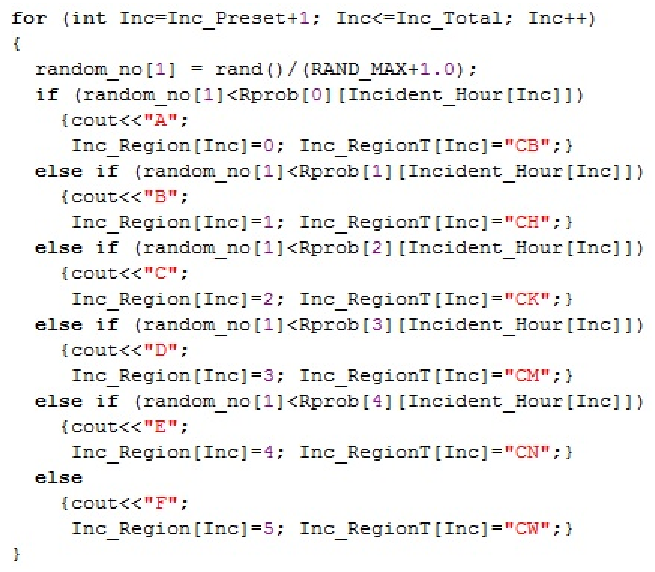
With the mean and standard deviation data that is input, a Poisson distribution is used to generate an amount of occurring incidents for each hour within the overall time period; note that the standard deviation is translated to variance in order to allow the standard form of Poisson distribution to be used. This distribution was selected as it is a fairly standard one that allows variability in data to be simulated using the information available here; namely mean and variance. These hourly rates of incidents are distributed across their respective hours through random assignation of starting times to take any minute value that falls within these hours. The C++ code used in order for this is represented in Figure 7.7. In this example a random real value between 0 and 1 is generated ('random_no[1]') which is then directly transformed into an integer value between 0 and 59 which is used to indicate the minute within the current hour at which the target incident initializes.



```
for (int Hour=0; Hour<24; Hour++)
{
  for (int Inc=1; Inc<=Incs_Start[Hour]; Inc++)
  {
    Inc_Total++;
    random_no[1] = rand()/(RAND_MAX+1.0);
    Incident_Start[Inc_Total]=int((Hour*60)+random_no[1]*60);
    Incident_Hour[Inc_Total]=Hour;
  }
}
```

Figure 7.7: Method by which individual incident starting times are generated.

The particular type and location of incident will however vary considerably and as such it is appropriate to allocate them varying durations of Police attendance. In order to classify each of the generated incidents a series of characteristics are required that define the region in which the incident occurs, the grade of the incident and the type of incident. For this, probabilities are calculated based upon the mean expected levels of each of the individual classifications occurring for each hour. For example, Figure 7.8 displays C++ code that was used to assign each generated incident a region of occurrence. A previously input array of values 'Rprob[X][Y]' contains for each hour Y the chance that an incident will occur in region X and it is from use of this that generation of a random value between 0 and 1 allows a region to be identified for each incident based upon the hour of occurrence. In this example, such a random value is generated and then compared against each of the cumulative probabilities defined to indicate each possible region of incident occurrence. A first test is performed comparing the generated real value against the first probability value. In the case that the generated value is less than the defined probability the incident is considered to have occurred within the associated region and the loop terminates. If the generated value is greater than the defined probability then the next probability is used to consider the second region. This process continues until all regions are considered; by this point the cumulative probability will have reached 1 and thus a region of occurrence will definitely have been defined for the incident.



```
for (int Inc=Inc_Preset+1; Inc<=Inc_Total; Inc++)
{
  random_no[1] = rand()/(RAND_MAX+1.0);
  if (random_no[1]<Rprob[0][Incident_Hour[Inc]])
    {cout<<"A";
     Inc_Region[Inc]=0; Inc_RegionT[Inc]="CB";}
  else if (random_no[1]<Rprob[1][Incident_Hour[Inc]])
    {cout<<"B";
     Inc_Region[Inc]=1; Inc_RegionT[Inc]="CH";}
  else if (random_no[1]<Rprob[2][Incident_Hour[Inc]])
    {cout<<"C";
     Inc_Region[Inc]=2; Inc_RegionT[Inc]="CK";}
  else if (random_no[1]<Rprob[3][Incident_Hour[Inc]])
    {cout<<"D";
     Inc_Region[Inc]=3; Inc_RegionT[Inc]="CM";}
  else if (random_no[1]<Rprob[4][Incident_Hour[Inc]])
    {cout<<"E";
     Inc_Region[Inc]=4; Inc_RegionT[Inc]="CN";}
  else
    {cout<<"F";
     Inc_Region[Inc]=5; Inc_RegionT[Inc]="CW";}
}
```

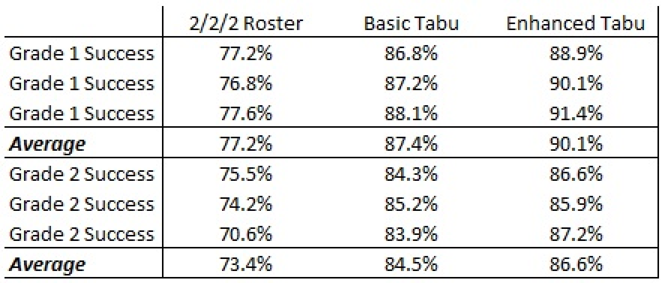
Figure 7.8: C++ code for random assignation of regions of occurrence for generated incidents.

Similar processes are used in assignation of a grade and type for each of the defined incidents which then provide a series of suitably well defined incidents. For each combination of region, grade and type of incident an array of values is input that describes the mean expected and variance of duration of such incidents by the hour of the day in which they initiated. A standard Poisson distribution is then used in order to generate, for each incident, a required amount of time in order to resolve it satisfactorily; this being the amount of time that a Police resource would be required in attendance. Through this the objective function may be implemented and each roster may be assessed through the generation of a pair of expected incident attendance success rates; one of these for grade 1 incidents and the other for grade 2 incidents.

7.3.2 Results from Tabu Search with Attendance Target Objective

In order to test this objective function, it was applied in conjunction with 2 different tabu search procedures; the first of these was through use of a tabu search algorithm with no enhancements and the second was tabu search with inclusion of diversification and intensification techniques. In order to provide a benchmark result against which the effectiveness of the tabu search approaches could be compared, the objective function was tested upon the standard roster currently employed by Leicestershire Police. The success rates for 3 repetitions of each of these trials along with their respective averages are included in Table 7.3 for both grade 1 and grade 2 incidents. Repeated trials for each of the 3 approaches were necessary due to the random nature of the incident generation process. Within Table 7.3 'Basic Tabu' refers to use of the tabu search methodology first investigated in which no search enhancement techniques are applied. 'Enhanced Tabu' refers to the final best found methodology, employing restart diversification and intensification in addition to multiple minima diversification. Tabu tenure penalties were each set to 5 as in the work conducted in Chapters 4 and 5 and the order of moves conducted was again staff redistribution followed by shift starting time adjustment and finally a secondary staff redistribution. The column titled '$2-2-2$ Roster' details the expected incident attendance success rates when considering application of the standard staffing roster as employed by Leicestershire Police.

Table 7.3: Expected incident attendance rates from the current Leicestershire roster and rosters resulting from tabu search.



	2/2/2 Roster	Basic Tabu	Enhanced Tabu
Grade 1 Success	77.2%	86.8%	88.9%
Grade 1 Success	76.8%	87.2%	90.1%
Grade 1 Success	77.6%	88.1%	91.4%
Average	77.2%	87.4%	90.1%
Grade 2 Success	75.5%	84.3%	86.6%
Grade 2 Success	74.2%	85.2%	85.9%
Grade 2 Success	70.6%	83.9%	87.2%
Average	73.4%	84.5%	86.6%

It is seen that the tabu search methodology previously proposed is also able to optimize staffing rosters with regards to this newly defined objective function, with increases seen in the quality of generated solutions beyond that provided from the initializing roster. Of note is the fact that attendance success rates for grade 2 incidents are commonly seen to be lower than the success rates of grade 1 incidents. This is caused by an additional component of the resource allocation decision making process in which grade 1 incidents are prioritized over grade 2 incidents. At the point a grade 1 incident occurs, if there are no immediately available resources for deployment then units currently in transit to an incident of lower importance are considered. Thus, if a unit is currently travelling to a grade 2 incident then they may instead be reassigned to attend the grade 1 incident with their previously assigned incident being marked as currently unattended and placed on the incident queue. The effects of this decision process are clearly illustrated within Table 7.3 with an anticipated reduced attendance rate at grade 2 incidents. The expected success rate could be improved through consideration of increased total staffing levels across the roster as a whole in the same way as applied within Sections 5.7, 5.8 and 5.9. It would be expected however that the elevated staffing levels this would suggest would be identical to those proposed with the previous objective function when considering the same trial problems. The reason for this is that the total volume of demand over time remains the same and as such would require the same total amount of staff in order to fully satisfy it. The freedom gained through providing incidents with a 'response window' as opposed to requiring immediate response could result in slight shifts to the effective amount of demand present at particular times of day; again this is unlikely to allow a reduction in the number of additional staff required.

This approach was only applied to the tabu search algorithm presented in Chapters 4 and 5 and as such there is potential to conduct further work in extension of the results provided through implementation using the genetic algorithm approach discussed in Chapter 6. The tabu search has shown itself to perform very strongly upon this type of combinatorial problem however and a strong improvement through application of genetic algorithms would not be an expected outcome. Indeed as the type of strings being generated by a genetic algorithm approach would not be changed for application with this new objective function it would be anticipated that it would suffer the same convergence problems as previously noted; this then leading to necessary division of the problem into a series of sub-problems. With no expected benefit beyond that provided by tabu search it is thus not worth applying the previously used genetic algorithm here.

7.4 Demand Profiling for Canine Trained Units

Within the Leicestershire Police organization there exist a large variety of officer types that are tailored to suit specific types of incident or other demand. It is of interest to investigate at least one such alternative officer type from the major front line officers already considered, with note being paid to the similarities and differences present. Such an alternative are the canine trained units that are used in attendance of high risk incidents that are also require rapid response.

7.4.1 Demands on Canine Trained Units

Demand as experienced by canine units is generated from the same sources as that for LPOs with the majority coming from either planned Police actions or in response to calls for action from the general public. The specific types of incidents that canine units will be required to attend differs however and there are several other key differences in the demands made upon their time.

7.4.1.1 Standard Staffing Policy Within this section of the Force there is, like in most other sections, a minimum staffing level set in order to encourage the meeting of a minimum performance level. This is simply set within the canine unit as the requirement that 2 active units be on duty at any point in time; ideally this would be higher but financial constraints within the force are a limiting factor. Each shift runs for 9 hours in length however only 8 hours of this will be operational duty as 30 minutes at the beginning and end of each shift are assigned in order to allow for feeding and exercise of the dogs in addition to travel to a Police premises to begin active duty. This travel time is essential as each dog will be living with their partnered officer outside of working hours and as such it is necessary for the pair to travel from their home location to a Police one in order to begin work.

7.4.1.2 Incident Attendance Canine units still attend incidents generated by calls from the general public (and thus those stored on the Police system OIS) however they are used more selectively, primarily attending incidents at which their presence will aid in reduction of risks. Accordingly, they are called out in response to more emergency incidents (grade 1) than any other type; this is supported by data collected by the canine unit in which it is stated that 70% of incidents they attend are grade 1, 28% are grade 2 and just 2% are grade 3.

In the case of most incidents attended by these units it will only be necessary that a single unit attend, however there are cases in which it would be desirable (if not essential) that at least 2 canine units are in attendance. For example it is considered to vastly reduce the risk to staff personnel involved at a firearms incident if 2 dog units attend instead of 1; though sometimes this is not possible, it should always be the case that 1 unit at least attends. This is a particularly strong constraint if the limited number of canine units that are planned to be available at any given time are considered. To help alleviate the pressure that this can cause, canine units may be released early from other incidents if their attendance is no longer required there.

7.4.1.3 Directed Action Policing Directed action policing is a key part of the duties surrounding the canine units, there are many pre-planned scenarios for which their attendance is a requirement. Most commonly occurring amongst these are football matches between County and National level teams for which a grading policy is used to classify the matches and assign an appropriate amount of canine units to manage the anticipated crowd levels if necessary. This policy is as follows:

Police Free	No canine units
Category A	No canine units
Category B	4 canine units
Category C	4 canine units
Category CIR	6 canine units

The categories ranging from A to C indicated increasing expected crowd population at such events with a further category of CIR representative of a category C match with anticipated increased risk. The impact of these and other such events is particularly high when the usual staffing pattern for these units is taken into consideration. Regularly only 2 or 3 units will be on duty at any time and it is further necessary to have at least 1 unit remain available in case another incident requiring a canine unit occurs at the same time.

7.4.1.4 Custodial Incidents As with most other Police incidents there is a chance that an arrest will result, for other Police departments this will then require further attendance of the arresting officer (and in some cases additional relieving staff) for up to 9 hours in order to fully process the arrestee. This is not the case for dog units however as due to their limited number it is considered more important to free them for return to duty as soon as possible. Thus in the case that a dog unit is involved with the arrest of an individual or group they will then hand over those arrested to another officer to handle the processing work.

7.4.1.5 Abstractions Abstractions are an area in which canine units will experience more restrictions upon availability than in other departments. The base cause for this is the fact that not only can an officer become ill or otherwise abstracted, so too can the dog with which they are partnered; this is known within the canine unit as double abstraction. Each pairing of Police officer and dog is a permanent thing, with the dog being housed with the officer outside of working hours. This is done to encourage a positive relationship to form between the two and indeed is known to be highly beneficial in the training and development of the canine units. From this then it can be understood that if a particular dog trained officer is abstracted from work then the dog with which he would normally work may not be passed to another officer. Likewise if a particular dog is abstracted, the handling officer partnered with them cannot then be assigned another dog to handle.

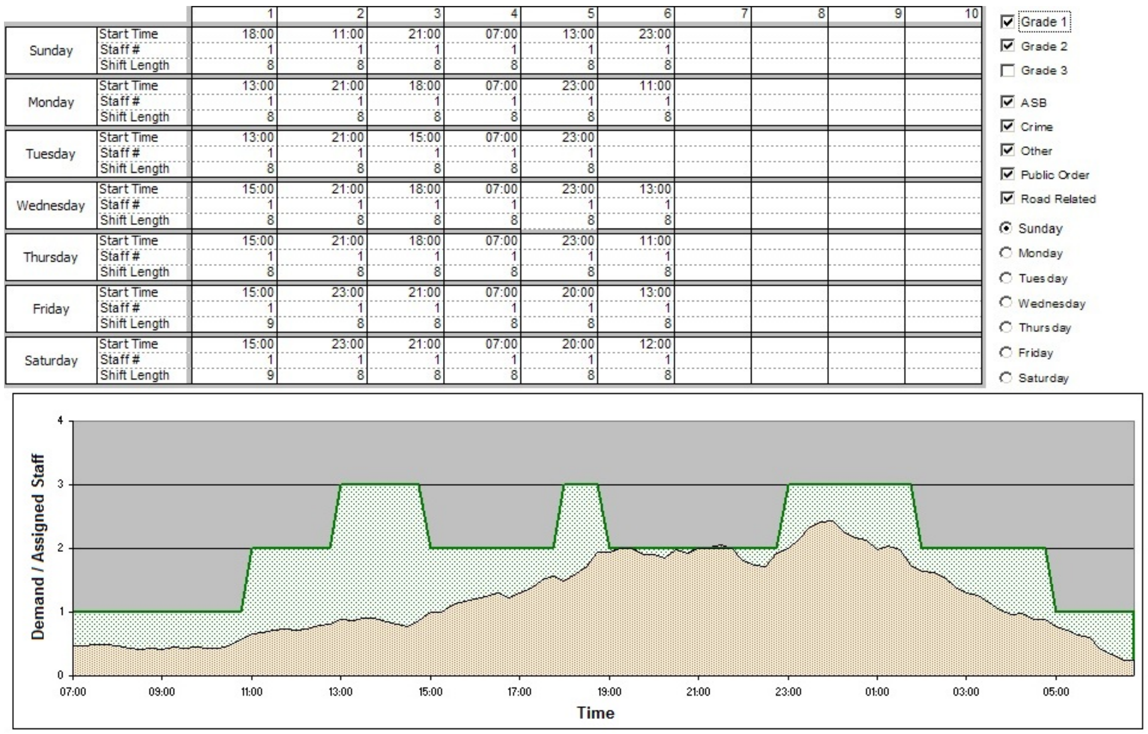
Clearly some of the abstractions that apply when considering a human Police officer will not be directly applicable to any of the dogs; for example maternity or annual leave. There is however additional abstraction time in the form of specialist training that needs to be considered, which will only affect the canine trained officers and their partnered dogs. There is a requirement for every single paired unit to attend a specific amount of training days per year which importantly takes the units out of Leicestershire Police premises and effectively abstracts them for the training period. The precise number of training days will vary based upon the specific dog type as follows:

General purpose dog	16 days
Drugs dog	10 days
Explosives dog	20 days

7.4.2 An Incident Demand Profiler Created for Canine Trained Units

The creation of an incident profiler tailored for canine trained units was a simplification of that already considered for frontline policing staff. With no requirement to consider custodial incidents being in requirement of an additional staff presence this factor was not necessary to include within the profiler. Indeed the additional factors of necessary training and attendance at specific events such as football matches were also not required to be automatically considered in compilation of expected demand as these are both examples of planned events for which additional staff beyond the standard requirement would be arranged. Thus the incident based demand occurring within any period of time for canine units is equal to all such incidents that occur within that period in addition to any that began previously and have yet to be fully resolved. This is directly comparable to the base incident data considered within the frontline policing staff incident based demand profiler.

A screenshot of the final profiling tool created using Microsoft Excel for canine trained units is included within Figure 7.9. As with the incident profiler demonstrated within Chapter 3 this tool provides an area for a user to define and distribute staff across a series of shifts. This is accomplished through entry of suitable values into the upper left table, where for example it is seen in the screenshot that a shift is defined on Sunday to start at 18:00 that lasts for 8 hours and is assigned a single member of staff. This user input staffing level is displayed alongside the expected level of incident demand requiring attendance of canine units in a graph at the base of the data entry table; allowing a user to immediately see whether the proposed shift pattern will be able to meet expected demand.



		1	2	3	4	5	6	7	8	9	10
Sunday	Start Time	18:00	11:00	21:00	07:00	13:00	23:00				
	Staff #	1	1	1	1	1	1				
	Shift Length	8	8	8	8	8	8				
Monday	Start Time	13:00	21:00	18:00	07:00	23:00	11:00				
	Staff #	1	1	1	1	1	1				
	Shift Length	8	8	8	8	8	8				
Tuesday	Start Time	13:00	21:00	15:00	07:00	23:00					
	Staff #	1	1	1	1	1					
	Shift Length	8	8	8	8	8					
Wednesday	Start Time	15:00	21:00	18:00	07:00	23:00	13:00				
	Staff #	1	1	1	1	1	1				
	Shift Length	8	8	8	8	8	8				
Thursday	Start Time	15:00	21:00	18:00	07:00	23:00	11:00				
	Staff #	1	1	1	1	1	1				
	Shift Length	8	8	8	8	8	8				
Friday	Start Time	15:00	23:00	21:00	07:00	20:00	13:00				
	Staff #	1	1	1	1	1	1				
	Shift Length	9	8	8	8	8	8				
Saturday	Start Time	15:00	23:00	21:00	07:00	20:00	12:00				
	Staff #	1	1	1	1	1	1				
	Shift Length	9	8	8	8	8	8				

☑ Grade 1
☑ Grade 2
☐ Grade 3

☑ ASB
☑ Crime
☑ Other
☑ Public Order
☑ Road Related

[✓ Sunday] Monday Tuesday Wednesday Thursday Friday Saturday

Figure 7.9: Screenshot of the canine unit incident profiling tool.
The upper left data input area allows for user input staffing allocations. On the graph displayed the orange region indicates the expected demand upon canine units whilst the green region is indicative of the defined supply of staff.

7.4.3 Summary

Through the creation of a demand profiling tool for canine trained units using the same base methodology as used in creation of the enhanced incident profiler for Local Police Officers (LPOs), it has been shown that the same approach may be suited to many departments within the Police organization that experience similar types of demand. There are many similarities between the demands present upon LPOs and those present upon canine trained units:

- Both experience demand borne from incidents that are automatically recorded by the same method in Police data records. With experience of the same data for the LPO incident profiler this allows for an immediate understanding of the base data available and allows a similar time based live incident estimation to be created.

- Special events such as football matches require the attendance of a certain number of both types of resource, dependent upon the scale and importance of the event.

- Both the canine units and LPOs used within Leicestershire are segregated into 5 working groups in order to facilitate the standard 2-2-2 shift pattern. Due to this the easiest adjustments to staffing rosters are those that may be achieved whilst retaining these working groups.

There are also several differences between the demands of LPOs and those of canine trained units:

- Canine units will not be engaged in any of the custody booking process in cases where an incident leads to arrest and will thus be much able to return to active duty and attend further incidents much more swiftly in these cases.
- For the purposes of sickness and other abstraction, each individual unit may be abstracted either due to the handler or the canine itself. Because of this, the abstraction per deployable unit (comprised of 1 dog and 1 handler) is higher than in other areas of the Force. Note that any pairing of handler and dog is unique and without a significant training period, a handler may not be paired with a different dog and nor may a dog be partnered with a different handler.
- There are a pair of 30 minute time periods within each rostered shift for canine units that is non operational time, dedicated to the feeding and exercise of dogs. Due to this, shifts of less than 9 hours should not be considered as they would lead to the creation of rosters that either have no shift overlap benefit or are simply incapable of meeting expected demand.

Due to the low volume of expected incidents and the low staffing numbers there would be little benefit to be obtained through implementation of either tabu search or genetic algorithms in optimization of staffing rosters here. For both ease of management in allocating staff to shifts and in provision of briefings at the start of shifts it would be desirable in such a small department to utilize 3 shift starting times within a day; defined using an early shift, a late shift and a night shift. Thus with a relatively low amount of potential variation in the available shift definitions and also in staffing allocations it would be expected that an exhaustive search of the solution space for this department could be conducted in a reasonable amount of time. If the department contained a greater number of available staff members this would allow more flexibility and would enable the consideration of additional shift definitions within a roster. The increase in scale of the solution space from this would make tabu search and genetic algorithms a viable option for staff roster optimization.

There is potential to apply this type of modelling practice to other departments within Leicestershire Police that experience similar quantifiable demand. Indeed any department in which response to incidents generated by calls to service from the general public are immediately identifiable as potential targets for this; examples of such departments include the firearms unit and the scientific support department. Of note the firearms unit are not called out to incidents with the same regularity as the members of other departments, however due to the severity of the incidents requiring their attendance there will always be a need to maintain such a unit ready for deployment. Through adaptation of the methodology used in quantification of demand it is possible to extend this work to create similar profiles for departments in which the base demand is in other forms. For example, by consideration of the expected arrival rate of arrestees at a custody suite over a period of time it would be possible to construct an estimation of the anticipated cell usage at that suite.

7.5 Practical Implementation of the Demand Profiler

7.5.1 Use of Incident Profiler in Assessment of Staff Rosters

The incident profiler has been utilized by Leicestershire Police in informing staffing decisions and considering options for long term adjustments to the basic staff rosters used. The specific rosters that are under consideration may not be discussed here due to their confidentiality, however generalized rosters may be discussed that share similarities with those that have been proposed.

The first roster necessary for consideration within any such work conducted within Leicestershire Police is the $2-2-2$ pattern currently employed. This roster provides a benchmark against which the benefits and drawbacks of any other proposed roster may be compared. This roster is precisely as previously defined and entails the deployment of staff across a rigidly defined early, late and night shift for each calendar day.

The second option proposed was a variable start pattern. This type of pattern is based upon the $2-2-2$ pattern with the addition that it allows variation in shift starting times and the division of any given shift into 2 or more shifts across which available staff are redistributed. This type of pattern is precisely of the form created as a result of tabu search implementation within Chapters 4 and 5 of this thesis. For illustration, Figure 7.10 displays a comparison between the $2-2-2$ roster and a variable shift pattern using the same total volume of staff for a single calendar day. The $2-2-2$ pattern is indicated by the thick blue line and the variable pattern is shown by the shaded green region.

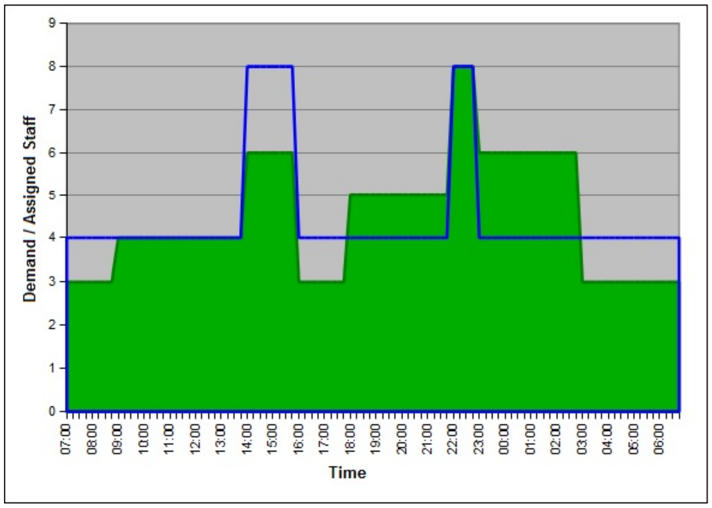


Figure 7.10: Comparison between $2-2-2$ roster and a variable shift pattern for a single day. The thick blue line describes the staffing provided by the $2-2-2$ roster and the green shaded region indicates that given by a variable shift pattern.

Both a 12 hour pattern and an 8 hour pattern were considered. These patterns result in a situation wherein shifts are placed back-to-back in order to cover the full 24 hour period of any given day. The result of this is that a consistent staffing allocation is maintained at all times across any given day. Indeed the work of this thesis strongly suggests that such shift patterns as these would be highly detrimental due to the likelihood that they will result in periods of understaffing and periods of overstaffing. Figure 7.11 illustrates a 12 hour shift pattern in comparison to the standard $2-2-2$ pattern; indeed an 8 hour pattern would follow the same flat trend however it would be at a lower level due to the reduced staffing allocation to each shift. The $2-2-2$ pattern is again illustrated by the thick blue line with the 12 hour pattern being shown by the shaded green region.

For the proposed rosters, the incident demand profiler was used in order to assess their capability to cater for the expected level of incident demand over the policing regions that they are intended to be implemented. With the option within the profiler to estimate elevation in demand it is possible to find the amount of cases in which the staff assigned to a given roster will be able to immediately respond to all occurring incidents. This is achieved through manual adjustment of the demand elevation level until the point at which the elevated demand is equal to the available supply for at least one point in time and lower than the available supply for all others. For example the graph displayed in Figure 7.12 displays such an eventuality with demand elevation set at 88%. This means that the roster would provide staffing that would be expected to be able to immediately respond to all occurring

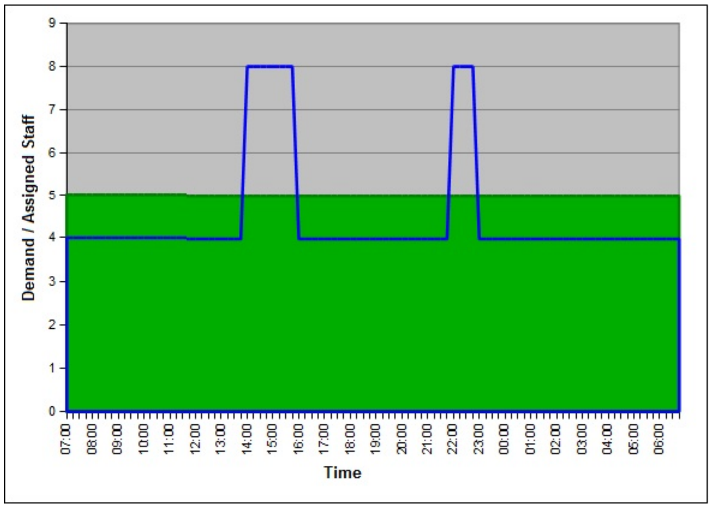


Figure 7.11: Comparison between $2-2-2$ roster and a 12 hour pattern for a single day. The thick blue line describes the staffing provided by the $2-2-2$ roster and the green shaded region indicates that given by a 12 hour shift pattern.

incidents in 88% of cases. If demand elevation is increased beyond this point, then demand outstrips supply between 19:45 and 21:00 and thus it may only be stated that this roster will be suitable to meet demand in 88% of cases and no more.

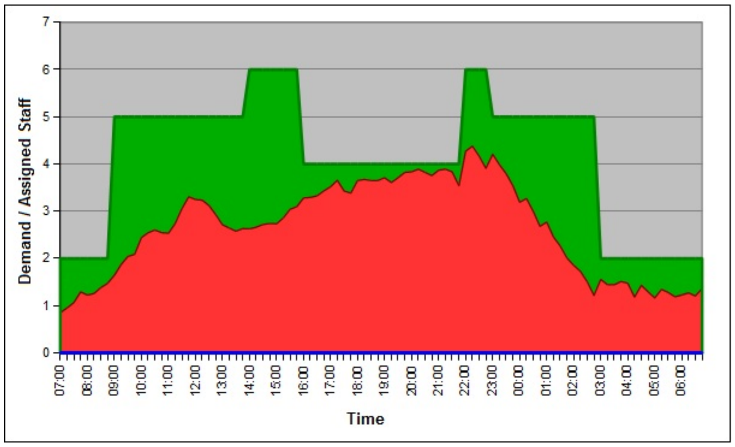


Figure 7.12: Graphical illustration of maximum demand elevation that may be met by a given roster.
The green shaded region describes allocated staffing whilst the red shaded region illustrates expected demand.

7.5.2 Abstraction for Front Desk Staff

In order to allow interaction between the Police and members of the public, the Leicestershire Force employ staff members to work at the front desks of local Police stations for public enquiries and assistance. This is a critical role and at no point may any of the front desks remain unstaffed. Due to this, there exist 2 major staffing groups that are used in order to ensure that staff are always available. These are the standard staff assigned to work at specific sites and a group of peripatetic staff that may work at any site in order to cover the shifts of the standard staff if they are unavailable to work. In the case that there are no peripatetic staff available to cover a given shift, it is then necessary to abstract either an LPO or PBO from their regular duty in order to provide the necessary front desk staffing. It was conjecturally stated that this eventuality is fairly common within Leicestershire and the operational impact of removing an active Policing resource at short notice can be very high.

Modelling the impact of staff abstraction upon these particular staff members then would provide an estimate of the amount of officers that would be required to be abstracted from active duty in order to cover the work of front desk staff. Further, through the consideration of roster adjustments for front desk and peripatetic staff it could be sought to minimize the amount of responsive officers required to cover for this role. The first step towards this goal is in provision of an estimate of the amount of shifts per year that front desk staff are abstracted from their work. To achieve this it is first necessary to collect data regarding the number of staff that work on front desks and the amount of shifts that each of them will work in any calendar year. Front desk staff are each assigned to 1 of 3 different types of site, that vary based upon their expected volume of business and the hours that they open. These are ranked gold, silver and bronze in descending order of business volume and are assigned staff as described in the table presented in Table 7.4. The columns of the table presented are headed to indicate which of the types of site the data they contain relates to; each column details the amount of staff at all sites of that type, the amount of days each staff member works and the total amount of days worked by all staff (this latter value being the product of the former 2).

The total annual working day values may then be used in order to estimate the amount of days that staff are expected to be abstracted and in requirement of cover. This is achieved through multiplication by a percentage indicative of the target abstraction rate. For example, the international standard abstraction rate when considering any staff planning is 30% and so for the gold sites detailed in Table 7.4 this would result in an expectation of 2187 $(0.3 *$

Table 7.4: Front desk staff assignation by type of site.

	Gold	Silver	Bronze
Staff	30	9	10
Working Days	243	243	260
Total Annual Days	7290	2187	2600

7290) abstracted shifts within a calendar year. The true abstraction rate seen within this department of Leicestershire Police is lower than this however and has been seen to steadily remain at approximately 21%. Table 7.5 describes the estimated numbers of abstracted days for each of these abstraction rates.

Table 7.5: Expected annual abstraction of front desk staff.

	Gold	Silver	Bronze
21% Abstraction	1531	459	520
30% Abstraction	2187	656	780

In order to assess whether the peripatetic staff should be expected to be able to regularly provide enough cover to meet all of these abstracted days, a similar calculation is required in order to estimate their total annual working days. Table 7.6 details the same information for peripatetic staff as was previously calculated for the front desk staff; this being the amount of staff employed in this role, the days each of them works annually, the total annual days worked by all staff and the amount of days expected to be abstracted.

Table 7.6: Expected annual working days for peripatetic staff.

Staff	9
Working Days	243
Total Annual Days	2187
21% Abstraction	459
30% Abstraction	656

From this, direct calculation may be performed to assess the difference between the amount of required shifts cover to be provided by peripatetic staff and the amount of shifts those staff are able to provide. Table 7.7 details this calculation through summation of the total expected abstracted days across all sites for each abstraction rate considered and then subtraction of the amount of cover days provided by peripatetic staff. This leaves a deficit number of shifts for which staff would need to be drawn from elsewhere within Leicestershire Police.

Table 7.7: Calculation of deficit shifts unmet by peripatetic staff.

	Gold	Silver	Bronze	Total	Cover	Deficit
21% Abstraction	1531	459	520	2510	1728	782
30% Abstraction	2187	656	780	3623	1531	2092

These results are in accordance with the conjectural statement that a large number of shifts are required to be covered by responsive staff members. Further, by consideration of 2 different abstraction rates and due to the linear nature of the relation between abstraction rate total abstracted shifts it is possible to present a graphical representation of these results. Such a representation is provided in Figure 7.13. On this graph the blue line indicates the amount of days expected to be abstracted as the abstraction rate varies whilst the pink line indicates the amount of peripatetic cover days provided as abstraction rate varies. The intersection of these 2 lines identifies the abstraction rate at which the amount of peripatetic cover days is equal to the amount required by front desk staff. Any abstraction rate lower than this will result in a surplus of cover whilst an abstraction rate higher than this will lead to a deficit; in this example the intersection occurs at an abstraction rate of approximately 16%. The yellow horizontal line included indicates the currently experienced situation in which abstraction is at a rate of 21%.

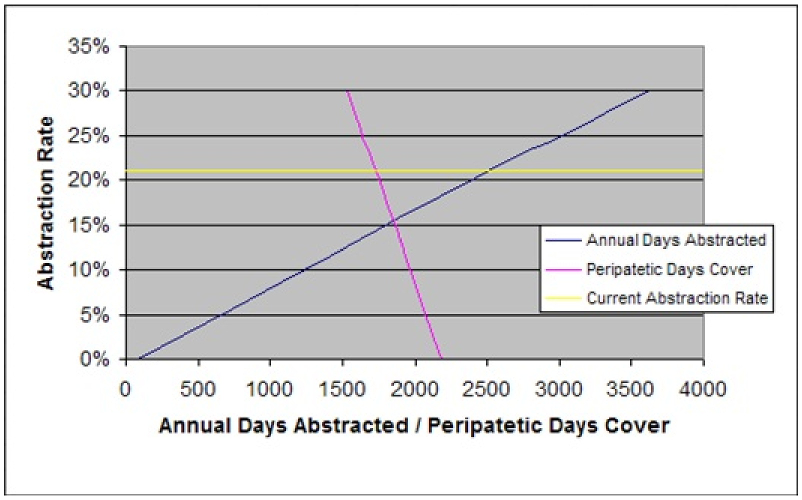


Figure 7.13: Graphical representation of peripatetic cover and front desk abstraction as abstraction rate changes.

Using this it is possible to consider either employment of additional peripatetic staff or the adjustment of their staffing rosters in order to increase the standard amount of working days. Indeed the latter of these options was undertaken by Leicestershire Police as a result from this work and standard annual working days for peripatetic staff were raised to 277. This results in a total annual number of 2493 working days, which reduces to 1969 after an abstraction rate of 21% is applied. Further considered were changes to the staffing numbers employed at front desks, resulting in the situation described by the values shown in Table 7.8. Though this does not entirely remove the problem, it has significantly reduced the expected numebr of shifts that will require cover from responsive Police staff; a reduction from 782 to 160 expected shifts needing cover annually when using an abstraction rate of 21%.

Table 7.8: Deficit peripatetic shifts after staffing level and staff roster adjustments.

	Gold	Silver	Bronze	Total	Cover	Deficit
Staff	20	9	10			
Working Days	260	260	260			
Total Annual Days	5200	2340	2600			
21% Abstraction	1092	491	546	2129	1969	160
30% Abstraction	1560	702	780	3042	1745	1404

7.5.3 Summary

This section has demonstrated the impact of the incident demand profiling tool within Leicestershire Police and has mentioned how it has been used to inform roster adjustment decisions for front line Police officers. Further an extension of the profiler to cater for front desk and peripatetic staff has been shown that has been applied in practice by Leicestershire Police; this resulting in the reduction of response officer abstraction to cover front desk shifts.

7.6 Hybridization

7.6.1 Introduction

For some optimization problems it may be seen that certain algorithms or techniques suffer in their ability to solve the problem due to some of their characteristics. For an example of this, the genetic algorithm included within the modeFRONTIER software presented in Section 6.3 was seen to struggle in solving the staff rostering problem posed by Leicestershire Police within a reasonable amount of time. One method available to overcome such situations is through consideration of a hybrid technique that utilizes elements of more than just a single search algorithm in order to compensate for any weaknesses within the original algorithm. This chapter will present such a hybridization through a combination of the modeFRONTIER genetic algorithm and the tabu search algorithm presented in Chapter 5. This algorithm was seen to become dominated by a strong solution within the initial population that was seen to propagate very well into later generations. Through this, final best found solutions using the algorithm contained many shifts with no staff assigned whatsoever. With the tabu search methodology used within this research designed to specifically take advantage of such shifts there is strong potential for a beneficial hybridization of the 2 techniques. The C++ based genetic algorithm however does not result in such a situation and as such it would not make a suitable hybridization partner for tabu search by the method proposed in this section.

It has been noted in the literature that genetic algorithms and tabu searches are highly complementary when used together to form a hybrid algorithm. In particular, Glover et al. [15] describe some potential methods by which the 2 distinct search methodologies may be unified in order to tackle vehicle routing problems. Though not of direct application to this research conducted regarding staff rostering, it does however highlight the fact that tabu search and genetic algorithms may be used in conjunction in order to enhance solution quality beyond that provided through each methodology when applied separately.

The potential strength of hybridization between genetic algorithms and tabu search is discussed within the work of Thangiah et al. [56] in which it is demonstrated that a particular hybrid algorithm is significantly stronger when applied to problems with known solutions than other popular metaheruistic techniques. Further diversity of the approach is illustrated in an application of a hybridization of the 2 base optimization algorithms to the problem of job scheduling in the work of Goncalves et al. [57]. In the paper it was proposed to use a genetic algorithm approach to first construct a series of suitable schedules and follow this with further solution improvement using a local neighbourhood search. This is precisely the style of technique that will be presented within this chapter and tested upon the same trial problems as previously used for both tabu search and genetic algorithms separately.

7.6.2 The Hybrid Search Algorithm

The hybridization proposed is more structurally simple than many of the methods proposed within the literature and seeks to use a tabu search algorithm in order to improve upon those results obtained through use of the modeFRONTIER genetic algorithm. The genetic algorithm used is seen to result in the generation of solutions that comprise a great many shifts with no assigned staff members. This is due to the relative strength of a particular member of the initial string population when compared to the other pseudo-randomly generated initial strings. With the genetic algorithm favouring the propagation of elements from strings that are stronger in comparison to the others it is no surprise that a particularly strong starting string is seen to greatly affect future generations. Usually mutation is able to compensate for this through random string variation, however as the zero staff shifts become more commonplace amongst generated strings it is seen that only a very high mutation rate would be able to impact the strings enough. This is not a suitable option however as high mutation rates lead to a mostly random search, with minimal tendency towards optimization due to the fact that the search will effectively be generating new strings entirely at random with each search iteration.

The tabu search created in Chapters 4 and 5 is seen to monopolize on precisely these problematic shifts, highly encouraging the distribution of staff members across shifts that were otherwise lowly staffed through use of a more methodical search process.

The hybridization method used within this research was to use the final best solutions generated using the modeFRONTIER genetic algorithm and input these as the system initialization point for the tabu search. This should be seen to improve the solution quality obtainable through use of either of the search methodologies individually. The genetic algorithm results would strongly benefit from the methodical staffing distribution provided as part of the tabu search algorithm whilst the tabu search itself will benefit through being initialized with a starting solution that is itself strong and yet has good potential for enhancement. A flow chart representative of this search process is displayed within Figure 7.14.

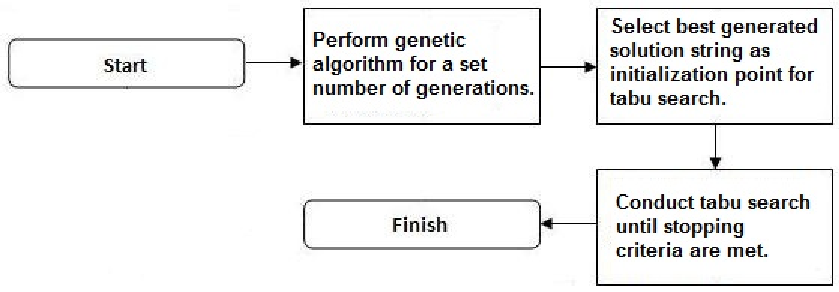

Start
Perform genetic algorithm for a set number of generations.
Select best generated solution string as initialization point for tabu search.
Conduct tabu search until stopping criteria are met.
Finish

Figure 7.14: A flow chart describing the hybrid search algorithm process.

Within this research the best performing of both algorithms were compiled in order to create the hybrid search algorithm. The genetic algorithm then used a population size of 32 strings and performed with a mutation rate of 7.5% until 100 new generations had been created. The tabu search method used a recency based teabu tenure penalty applied to solution objective function scores of 5 and made us of restart diversification, multiple minima diversification and restart intensification.

7.6.3 Results

The described methodology was conducted a total of 15 times, the results from which are displayed in Table 7.9 alongside results from conducting the tabu search method with the default starting solution. The columns GA1 to GA5 represent 5 unique genetic algorithm searches conducted using modeFRONTIER with the system criteria described. The first row 'Pre Tabu Score' indicates the solution quality of the best result found by the genetic algorithm; this result being that which becomes the initializing solution for the ensuing tabu search. This generated solution was then used as the initialization point for 3 tabu searches, each of which uses the exact same system criteria. Such action was performed in order to allow any anomalous results created through the semi-random nature of both search methodologies to be easily identified; no particular solutions stand out in these results as anomalous and thus each of them is representative of the cap

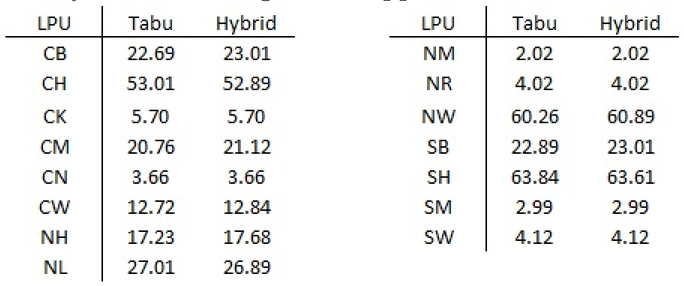
Table 7.9: Results from application of the hybrid search algorithm.

	GA 1	GA 2	GA 3	GA 4	GA 5	Tabu Alone
Pre Tabu Score	220.15	214.68	195.37	208.69	232.20	
Tabu Search 1	24.72	24.34	25.04	24.13	24.83	24.51
Tabu search 2	24.05	24.64	24.51	24.39	24.07	23.90
Tabu Search 3	24.68	24.51	24.16	24.03	24.69	24.12

Resulting solution quality from use of the hybrid method is similar to that provided through use of the tabu search methodology alone. Due to this it is possible that the variation in the tabu search initializing solution provided through use of the genetic algorithm has little actual impact on final solution quality; meaning that the tabu search is strong enough to reach the same resulting solution quality regardless of the inclusion of search initialization through use of the genetic algorithm. In order to verify this claim, the same hybrid method was applied to each of the 15 LPU based trial problems with the results from this illustrated in Table 7.10. Within these results it is seen that the hybrid method is able to outperform the tabu search method but does not do so consistently enough for this to be more than the result of random variation due to the random elements of the respective search methodologies.

The final generated solution rosters in each of the trial problems are strongly similar to each other for both the tabu search alone and for the hybridization method. Indeed it is

Table 7.10: Results of hybrid search algorithm application to 15 LPU based trial problems.



LPU	Tabu	Hybrid
CB	22.69	23.01
CH	53.01	52.89
CK	5.70	5.70
CM	20.76	21.12
CN	3.66	3.66
CW	12.72	12.84
NH	17.23	17.68
NL	27.01	26.89
NM	2.02	2.02
NR	4.02	4.02
NW	60.26	60.89
SB	22.89	23.01
SH	63.84	63.61
SM	2.99	2.99
SW	4.12	4.12

commonly seen that across the full week rosters only a few staff members will be differently allocated and the defined shifts themselves will be identical. The reason for slight difference in the final solutions could be due to different paths being taken towards optimization from different initializing solutions; this is particularly encouraged through application of restart diversification within the tabu search. Thus as near optimality is reached, the different moves used to reach the same region of the solution space will result in differing paths to optimization and thus different elements being penalized by tabu tenure penalties. When near optimality the number of improving moves available within the neighbourhood of the current solution will be relatively low and thus these penalties have greater potential to act in a restrictive manner. Thus it is likely that such situations as demonstrated will arise in which final generated solutions differ only by the placement of a few staff members, where the penalty to selection of the associated search elements is far greater than the very minor improvement to solution quality their selection would afford.

The search parameters of each individual method have been varied widely in order to find those that result in the strongest performance. Due to this, their variation within this search would be expected to yield no improvement to the quality of solutions generated. Indeed parameter variation within a single search routine would only be seen to weaken the effectiveness of it with no anticipated benefit provided to the other routine. If however the search were acting in a manner whereby it is seen to constantly switch between the 2 available search algorithms then it could be of benefit to lower the strength of 1 in order to allow the other a more active role in the optimization. This presents an interesting potential direction for further research in the construction of such an alternating search algorithm. With tabu search being an intensifying algorithm by design it is possible that intermittent use of genetic algorithms throughout the search could provide a wider range of diversification leading to higher quality solutions.

7.6.4 Summary

Through the work presented in this section it has been shown that a hybrid search algorithm comprising a preliminary genetic algorithm search which is then continued through use of tabu search is able to solve the staff rostering problem considered with generation of high quality solutions. There remains the possibility however that the initialization solutions provided through genetic algorithm are not of strong benefit to the overall algorithm. Indeed with similar solution quality afforded through both the hybrid methodology and the tabu search when conducted alone, it is at least possible to use the non-hybrid tabu search and reach suitable solution quality with less computation time. It is concluded that though the hybrid approach performs well, it would be recommended that the tailored tabu search method be used instead for the target problem.

7.7 Chapter Summary

This chapter has presented a variety of additional work conducted as part of the research for this thesis. In summary:

- A methodology by which incidents may be defined probabilistically through the assignation of key characteristics that dictate the expected level of resourcing required for their resolution has been demonstrated. This provides an alternative method whereby random variation in incident based demand may be assessed.

- An alternative objective function was proposed that considered incident attendance rate targets as a metric for success. This is a commonly used performance indicator within many Police organizations within the UK. Emphasis within Leicestershire has however focussed upon potential cost savings and reinvestment within other areas of the business that may be obtained through staffing efficiency. Thus it has been more appropriate within this work to utilize the first objective function proposed.

- The canine trained units within Leicestershire Police were discussed with the creation of an adapted incident profiler that was tailored to suit their department. The use of operational research techniques was noted as being expected to provide little benefit to this department due to the low complexity of the staff rostering problem that would be created.

- Practical implementations of the incident demand profiler within Leicestershire Police have been discussed. Specifically its use in informing permanent staff roster adjustments and the consideration of abstraction for front desk staff have been presented.

- Hybridization of the best performing tabu search methodology with the genetic algorithm provided within modeFRONTIER was conducted. It was seen the hybrid algorithm is capable of generation solutions of a similar quality to the tabu search, however it does so with a much greater calculation cost and thus it is not a recommended approach for the problem considered within this thesis.

Chapter 8. Conclusions and Future Work

8.1 Introduction

In this thesis a model for the demands present upon Local Police Officers (LPOs) has been created that allows a representation to be compiled indicative of the required number of staff to meet all such demand at any point in time. This model has been used in definition of a series of trial problems through the coupling of generated profiles of demand with appropriate staffing rosters. Both a tabu search metaheuristic and genetic algorithms have been applied to the defined staff rostering problems through the use of C++ and have been shown able to yield final solution rosters of a similar quality.

8.2 Summary

Within this thesis, several major tasks have been undertaken and completed:

- The work of other researchers in the field of demand modelling within other industries was investigated with particular attention paid to methodologies employed. Further, a selection of optimizing search algorithms were reviewed that presented some potential application within this research. From this 2 particular methodologies were selected for further consideration and eventual implementation within this thesis; these being tabu search and genetic algorithms.

- A review and evaluation of the demands present upon front line staff within the Police organization of Leicestershire was performed, leading to a full understanding of the types of demand affecting these staff members and the required response necessary in order to satisfy them.

- A new incident based demand profiling tool was created for use in assessing the ability of a given staffing roster to meet expected levels of demand. This tool was used in the definition of a series of trial problems that were later used in order to test the operational research techniques considered.

- A tabu search optimization approach was applied to the defined trial problems with investigation into the effects of variation in key search parameters. On discovery of well suited search parameters, diversification and intensification enhancements were constructed and applied in conjunction with the best performing tabu search in order to further enhance solution quality.

- Genetic algorithms were applied to the same series of trial problems in order to provide comparison against the results obtained through tabu search. Both a commercial software package and a custom constructed C++ approach were tested. The commercial software was seen to suffer from very slow convergence; this problem was solved through use of the C++ program coupled with redefinition of the trial problems. In both cases, suitable search parameters were discovered through trial variation.

- A hybridization of tabu search with genetic algorithms was considered and 1 such methodology was applied to the trial problems. This was seen to provide an improvement to the solutions generated through genetic algorithms alone when applied to the original trial problems, however it did not strongly improve upon results obtained using tabu search.

- The demand modelling work was expanded to consider a further department within Leicestershire Police; this department being the canine trained units.

8.3 Conclusions

The final model created for LPO demand takes account of the following factors:

- Probabilistic variation in the number of expected incidents through consideration of variation in historical data.

- Expected duration of incidents generated through calculation of a mean average from 2 years of historically recorded data.

- The effects of double crewing, this being the situation in which multiple Police officers work as a single unit for the duration of their shift in order to increase their security.

- Incidents that result in an arrest and thus require an amount of additional staff time in order to resolve satisfactorily.

- Additional administrative duties that may be present for any particular type of incident beyond that which would otherwise be recorded in historical data.

- Variation in incident duration and occurrence rates through consideration of the 5 major incident types and 4 incident grades. The 5 types are anti-social behaviour, crime, public safety, road related and other whilst the 4 grades are used to indicate severity of the incident.

Through application of the tabu search and genetic algorithms high quality solutions for the defined staff rostering problems have been generated. The best performing tabu search methodology was shown to:

- Employ a multiple minima diversification to ensure that a range of solutions within the neighbourhood of the first found are investigated by the search. This helps to discourage the search becoming trapped by a strong local attractor and encourages the search to find results which are more likely to be global optima.

- Use a restart diversification modification in perturbation of the shift starting times defined within the initial solution. This modification is informed by a previously conducted search and identifies those shifts that were least selected for adjustment. Perturbing these shifts specifically at initialization will force the next search into an entirely new region of the solution space.

- Include a restart intensification that performs a preliminary search on a subset of the available search elements that aims to focus upon a particularly lucrative region of the solution space before initializing the full tabu search. The benefit of this particular inclusion was fairly low, however the tabu search is known to be an intensifying search and the effects of further intensification are expected to be small.

- Use 3 different moves and apply these in a particular sequence. This sequence was to first distribute staff across shifts defined in the initializing solution, then to trial increase of the starting times of these shifts followed by the third move type attempting decrease of the starting times. Finally a secondary staff redistribution is implemented in order to allow the staff distribution to compensate for the newly redefined shift structure. The moves are defined in Section 4.4.2 and this sequence is equivalent to performing moves of type 1, type 2, type 3 and finally type 1 again.

- Use recency based tabu tenure penalties based upon how recently any particular element was last selected to conduct a move within the search process. The most commonly seen well suited tenure penalty was 5, this meaning that used elements will remain penalized for the following 5 search iterations.

- An ability to adjust the total staffing level available across the roster as a whole coupled with methodical shift length variation. Allowing the search to find the least number of staff required across the most optimally defined shifts in order to meet all expected demand.

The most effective application of genetic algorithms was achieved through use of C++ and considered the following:

- Mutation of the staffing level string elements at creation of new solution strings with a selection rate of 5%. This encourages diversification within the search and indeed allows staff level values outside of those included within the original string population to be considered.
- Other elements of the search strings were not mutated as this was seen to be highly detrimental to the search. The effects of allowing random mutation in shift starting times and durations was too impactive upon the search and resulted in generation of much weaker solution strings, damaging the ability of the search to find improving solutions.
- Through redefinition of the base problem from a single weekly representation of demand and an associated staffing roster into a series of 7 daily problems it was possible to generate solutions of similar quality to the tabu search. Convergence of the search towards a suitably high quality result was very slow when considering the full week as a single problem.
- Solutions were identified as valid or invalid by consideration of the total staffing allocation that they represent across a full roster. Invalid solutions were heavily penalized to discourage their future selection within the breeding process.

The field of operational research has been added to with this research through use of the tabu search methodology to the problem defined coupled with implementation of diversification and intensification techniques defined by the author. This has been recognized by the Journal of the Operational Research Society and a paper detailing some of this work has been peer reviewed and accepted for publication under the title 'A Tabu Search Algorithm Applied to the Staffing Roster Problem of Leicestershire Police Force'. The incident demand profiling tool constructed within this research was presented at Simulation Workshop 2010 alongside a peer reviewed paper titled 'An Application of Demand Profiling and Optimization of Staffing Levels within Leicestershire Police Force' that appeared in the proceedings of the conference. Indeed the staff rostering problem itself is unique in that there is very little applied research towards the optimized rostering of Police staff members. In contrast the nurse rostering problem is very highly researched but as has been noted, there are several differences between that problem and the one posed within this thesis.

8.4 Future work

There are many potential ways by which this research could be extended. These are based around 2 key categorizations, the consideration of additional factors that allow a more complete image of demand to be created and review of application within Leicestershire Police with further implementation and development in the industrial setting.

- To enhance the realism of the demand model it would be of benefit to obtain realistic data for input relating to abstraction rates, administration time, unit crewing policy and incident attendance policies. This would result in the model being tailored specifically to suit the Leicestershire Police organization by considering precisely their policies and situation.

- Expansion of the research to consider other areas and departments within the Force. Of particular interest are Police Community Support Officers, Criminal Investigation Department and Police Beat Officers. There is further potential to such an expansion as it would allow the effects of interactions between departments and officers to be considered.

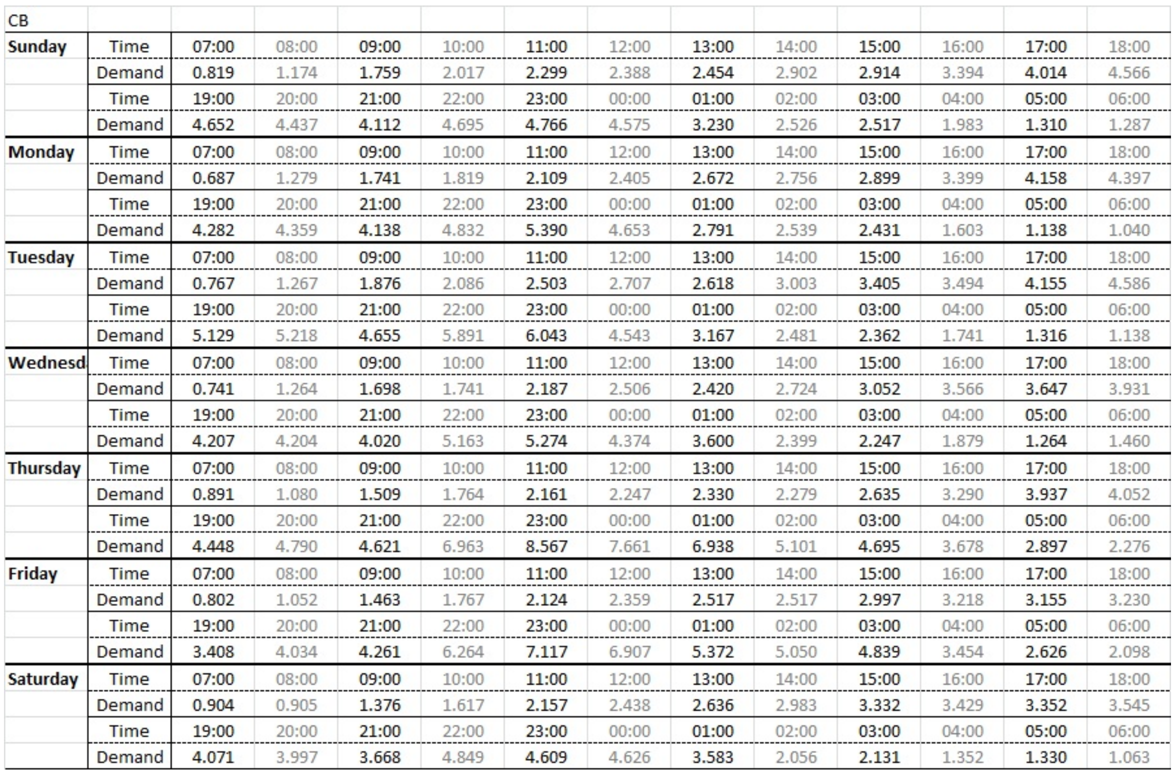
- Strong benefit could be gained through consideration of the working conditions of staff members. This would result in a model that is able to quantify the expected level of staff satisfaction with any given roster for each individual Officer (thus also allowing an average staff satisfaction to also be defined). Through implementation of the proven optimization techniques of tabu search and genetic algorithms, rosters could then be created that meet demand and also provide high staff satisfaction.

- Considering geographic factors could provide a benefit within future work. Indeed there is potential to optimize incident response times through consideration of the best placement of Officer waymarkers (key geographic points at which Police staff will wait until an incident occurs).

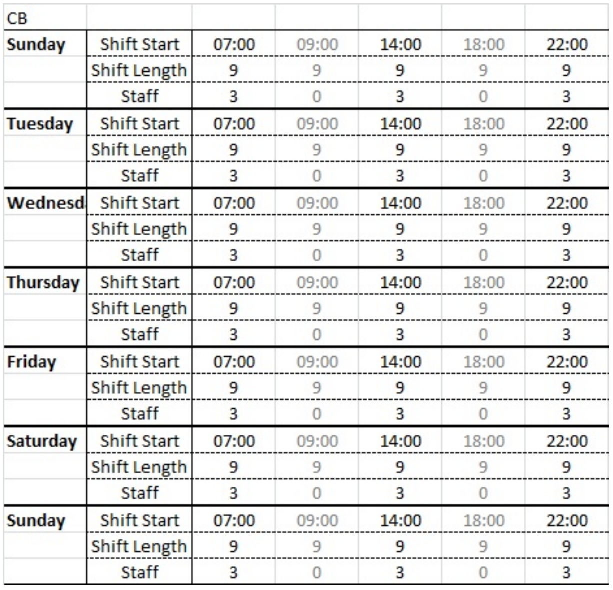
- Investigation of potential seasonal fluctuations in demand and the effects of specific events such as bank holidays or local events would allow more specific staff rostering problems to be created and understood. Indeed with a full understanding of any such demand variation a more robust model of the demands on Police staff could be created.

- Through contact with the Police organizations of other Counties it would be possible to expand the research to better suit a more diverse range of Police staffing problems than those posed by the Leicestershire Force alone.

Appendix A. Trial Problems.

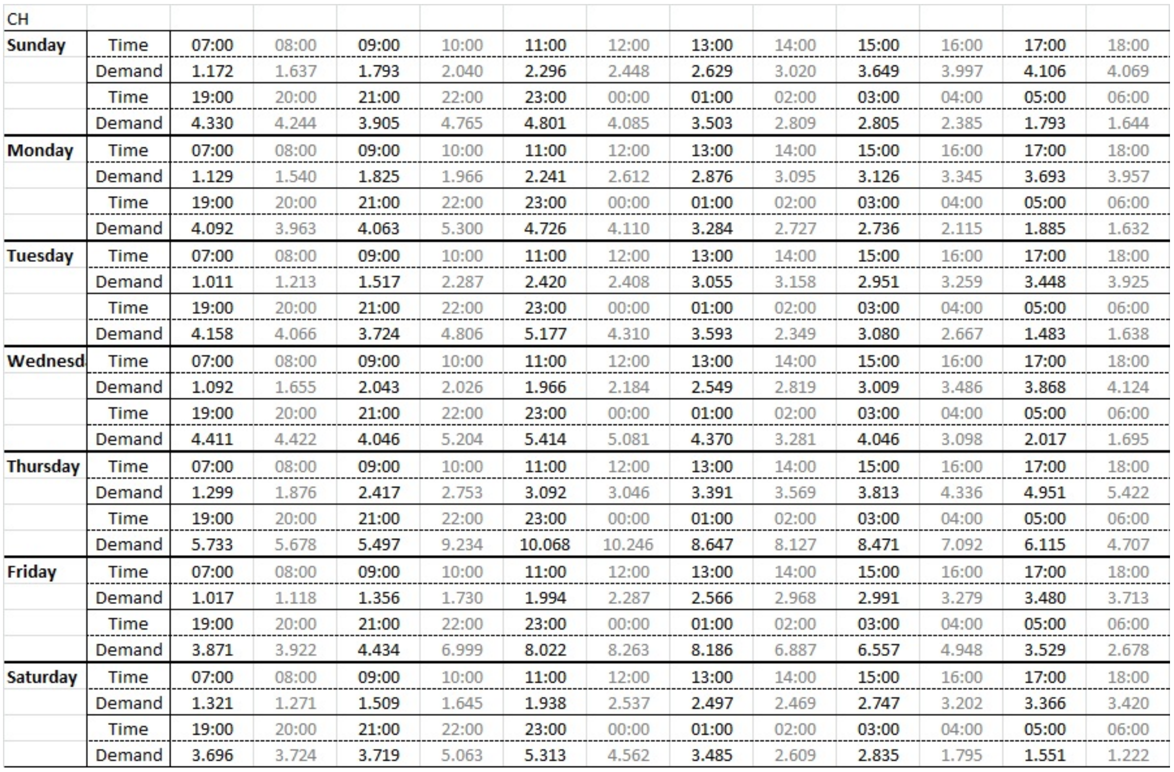


CB													
Sunday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.819	1.174	1.759	2.017	2.299	2.388	2.454	2.902	2.914	3.394	4.014	4.566
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	4.652	4.437	4.112	4.695	4.766	4.575	3.230	2.526	2.517	1.983	1.310	1.287
Monday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.687	1.279	1.741	1.819	2.109	2.405	2.672	2.756	2.899	3.399	4.158	4.397
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	4.282	4.359	4.138	4.832	5.390	4.653	2.791	2.539	2.431	1.603	1.138	1.040
Tuesday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.767	1.267	1.876	2.086	2.503	2.707	2.618	3.003	3.405	3.494	4.155	4.586
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	5.129	5.218	4.655	5.891	6.043	4.543	3.167	2.481	2.362	1.741	1.316	1.138
Wednesd	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.741	1.264	1.698	1.741	2.187	2.506	2.420	2.724	3.052	3.566	3.647	3.931
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	4.207	4.204	4.020	5.163	5.274	4.374	3.600	2.399	2.247	1.879	1.264	1.460
Thursday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.891	1.080	1.509	1.764	2.161	2.247	2.330	2.279	2.635	3.290	3.937	4.052
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	4.448	4.790	4.621	6.963	8.567	7.661	6.938	5.101	4.695	3.678	2.897	2.276
Friday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.802	1.052	1.463	1.767	2.124	2.359	2.517	2.517	2.997	3.218	3.155	3.230
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.408	4.034	4.261	6.264	7.117	6.907	5.372	5.050	4.839	3.454	2.626	2.098
Saturday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.904	0.905	1.376	1.617	2.157	2.438	2.636	2.983	3.332	3.429	3.352	3.545
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	4.071	3.997	3.668	4.849	4.609	4.626	3.583	2.056	2.131	1.352	1.330	1.063

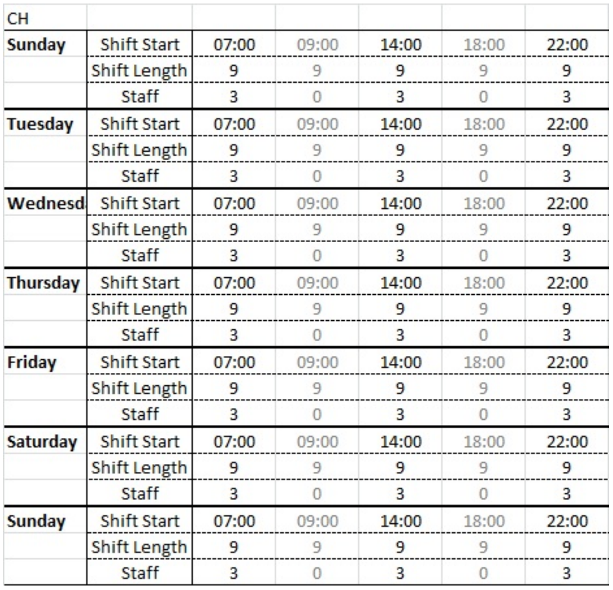


CB						
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	3	0	3	0	3
Tuesday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	3	0	3	0	3
Wednesd	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	3	0	3	0	3
Thursday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	3	0	3	0	3
Friday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	3	0	3	0	3
Saturday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	3	0	3	0	3
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	3	0	3	0	3

Figure A-1: Trial problem based upon the CB region of Leicestershire.

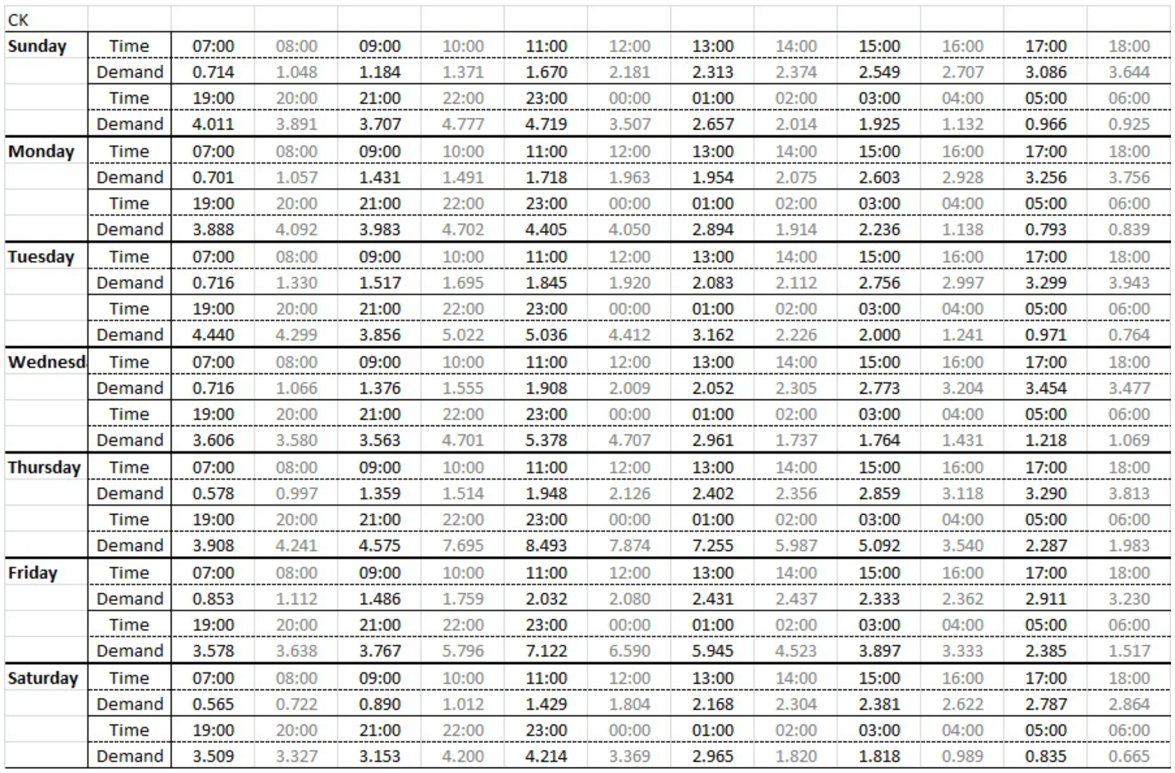


CH													
Sunday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	1.172	1.637	1.793	2.040	2.296	2.448	2.629	3.020	3.649	3.997	4.106	4.069
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	4.330	4.244	3.905	4.765	4.801	4.085	3.503	2.809	2.805	2.385	1.793	1.644
Monday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	1.129	1.540	1.825	1.966	2.241	2.612	2.876	3.095	3.126	3.345	3.693	3.957
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	4.092	3.963	4.063	5.300	4.726	4.110	3.284	2.727	2.736	2.115	1.885	1.632
Tuesday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	1.011	1.213	1.517	2.287	2.420	2.408	3.055	3.158	2.951	3.259	3.448	3.925
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	4.158	4.066	3.724	4.806	5.177	4.310	3.593	2.349	3.080	2.667	1.483	1.638
Wednesd	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	1.092	1.655	2.043	2.026	1.966	2.184	2.549	2.819	3.009	3.486	3.868	4.124
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	4.411	4.422	4.046	5.204	5.414	5.081	4.370	3.281	4.046	3.098	2.017	1.695
Thursday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	1.299	1.876	2.417	2.753	3.092	3.046	3.391	3.569	3.813	4.336	4.951	5.422
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	5.733	5.678	5.497	9.234	10.068	10.246	8.647	8.127	8.471	7.092	6.115	4.707
Friday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	1.017	1.118	1.356	1.730	1.994	2.287	2.566	2.968	2.991	3.279	3.480	3.713
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.871	3.922	4.434	6.999	8.022	8.263	8.186	6.887	6.557	4.948	3.529	2.678
Saturday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	1.321	1.271	1.509	1.645	1.938	2.537	2.497	2.469	2.747	3.202	3.366	3.420
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.696	3.724	3.719	5.063	5.313	4.562	3.485	2.609	2.835	1.795	1.551	1.222

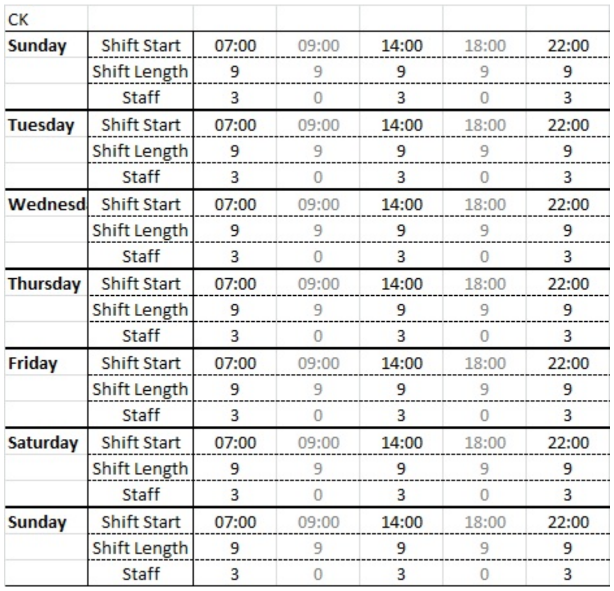


CH						
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	3	0	3	0	3
Tuesday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	3	0	3	0	3
Wednesd	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	3	0	3	0	3
Thursday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	3	0	3	0	3
Friday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	3	0	3	0	3
Saturday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	3	0	3	0	3
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	3	0	3	0	3

Figure A-2: Trial problem based upon the CH region of Leicestershire.

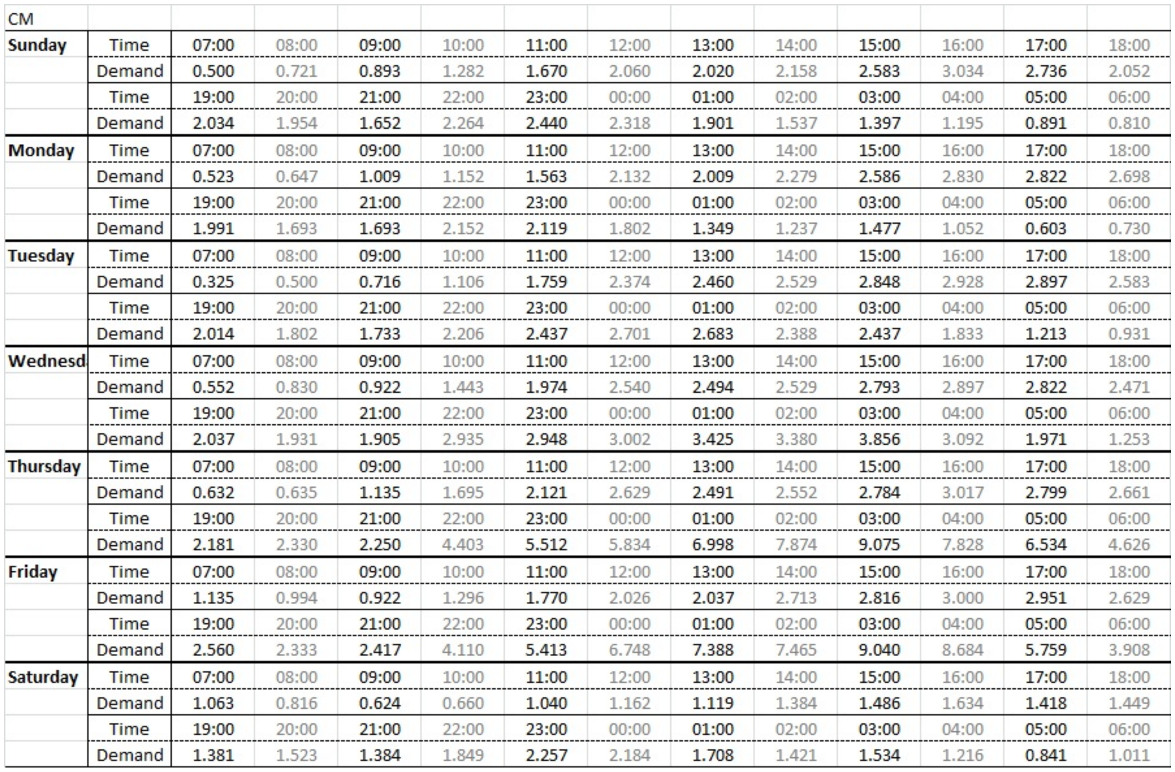


CK													
Sunday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.714	1.048	1.184	1.371	1.670	2.181	2.313	2.374	2.549	2.707	3.086	3.644
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	4.011	3.891	3.707	4.777	4.719	3.507	2.657	2.014	1.925	1.132	0.966	0.925
Monday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.701	1.057	1.431	1.491	1.718	1.963	1.954	2.075	2.603	2.928	3.256	3.756
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.888	4.092	3.983	4.702	4.405	4.050	2.894	1.914	2.236	1.138	0.793	0.839
Tuesday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.716	1.330	1.517	1.695	1.845	1.920	2.083	2.112	2.756	2.997	3.299	3.943
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	4.440	4.299	3.856	5.022	5.036	4.412	3.162	2.226	2.000	1.241	0.971	0.764
Wednesd	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.716	1.066	1.376	1.555	1.908	2.009	2.052	2.305	2.773	3.204	3.454	3.477
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.606	3.580	3.563	4.701	5.378	4.707	2.961	1.737	1.764	1.431	1.218	1.069
Thursday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.578	0.997	1.359	1.514	1.948	2.126	2.402	2.356	2.859	3.118	3.290	3.813
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.908	4.241	4.575	7.695	8.493	7.874	7.255	5.987	5.092	3.540	2.287	1.983
Friday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.853	1.112	1.486	1.759	2.032	2.080	2.431	2.437	2.333	2.362	2.911	3.230
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.578	3.638	3.767	5.796	7.122	6.590	5.945	4.523	3.897	3.333	2.385	1.517
Saturday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.565	0.722	0.890	1.012	1.429	1.804	2.168	2.304	2.381	2.622	2.787	2.864
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.509	3.327	3.153	4.200	4.214	3.369	2.965	1.820	1.818	0.989	0.835	0.665

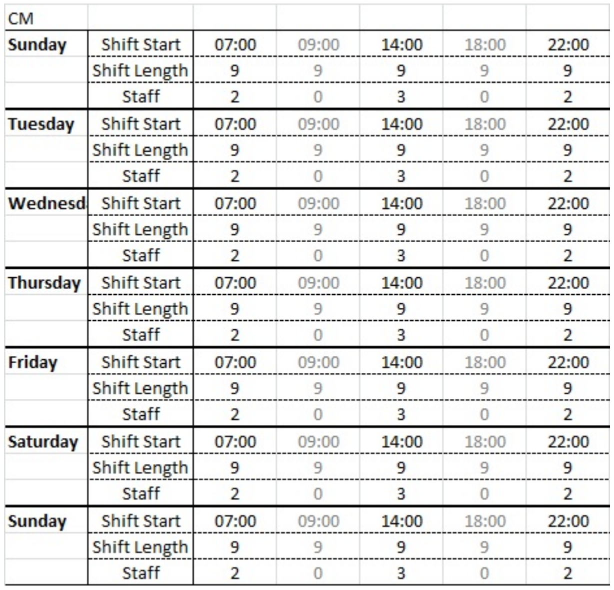


CK						
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	3	0	3	0	3
Tuesday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	3	0	3	0	3
Wednesd	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	3	0	3	0	3
Thursday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	3	0	3	0	3
Friday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	3	0	3	0	3
Saturday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	3	0	3	0	3
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	3	0	3	0	3

Figure A-3: Trial problem based upon the CK region of Leicestershire.

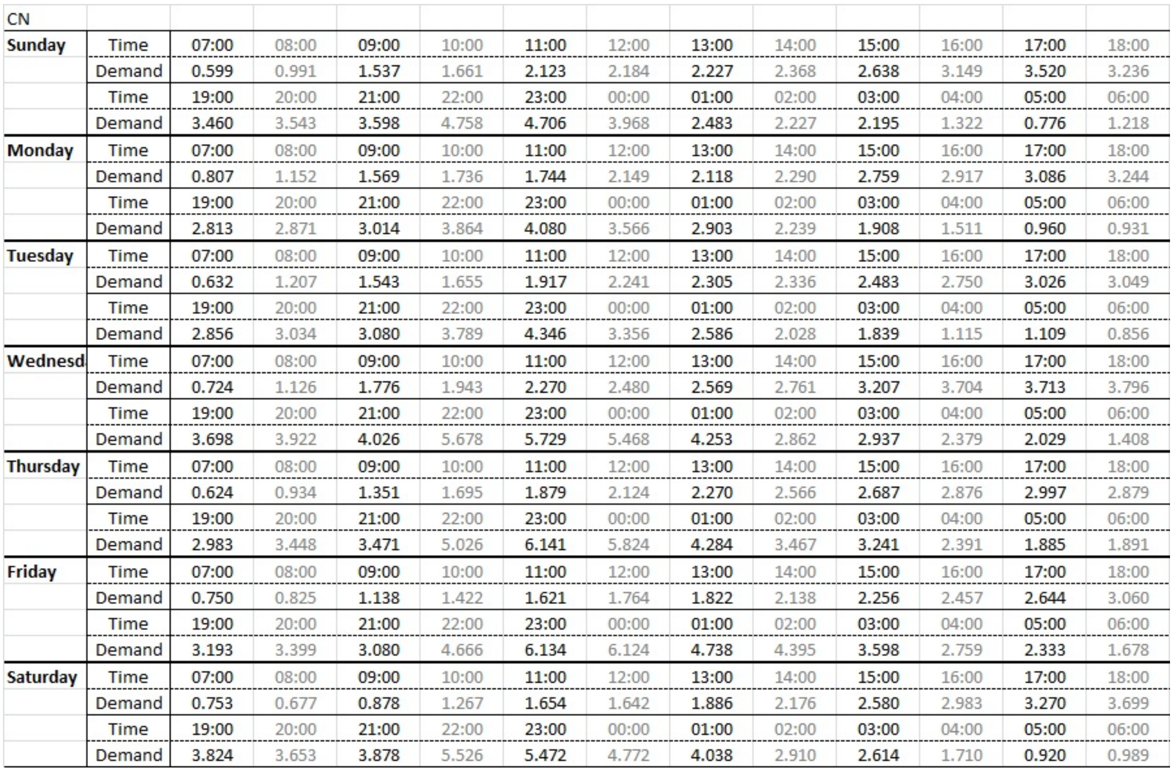


CM													
Sunday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.500	0.721	0.893	1.282	1.670	2.060	2.020	2.158	2.583	3.034	2.736	2.052
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	2.034	1.954	1.652	2.264	2.440	2.318	1.901	1.537	1.397	1.195	0.891	0.810
Monday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.523	0.647	1.009	1.152	1.563	2.132	2.009	2.279	2.586	2.830	2.822	2.698
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	1.991	1.693	1.693	2.152	2.119	1.802	1.349	1.237	1.477	1.052	0.603	0.730
Tuesday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.325	0.500	0.716	1.106	1.759	2.374	2.460	2.529	2.848	2.928	2.897	2.583
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	2.014	1.802	1.733	2.206	2.437	2.701	2.683	2.388	2.437	1.833	1.213	0.931
Wednesd	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.552	0.830	0.922	1.443	1.974	2.540	2.494	2.529	2.793	2.897	2.822	2.471
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	2.037	1.931	1.905	2.935	2.948	3.002	3.425	3.380	3.856	3.092	1.971	1.253
Thursday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.632	0.635	1.135	1.695	2.121	2.629	2.491	2.552	2.784	3.017	2.799	2.661
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	2.181	2.330	2.250	4.403	5.512	5.834	6.998	7.874	9.075	7.828	6.534	4.626
Friday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	1.135	0.994	0.922	1.296	1.770	2.026	2.037	2.713	2.816	3.000	2.951	2.629
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	2.560	2.333	2.417	4.110	5.413	6.748	7.388	7.465	9.040	8.684	5.759	3.908
Saturday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	1.063	0.816	0.624	0.660	1.040	1.162	1.119	1.384	1.486	1.634	1.418	1.449
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	1.381	1.523	1.384	1.849	2.257	2.184	1.708	1.421	1.534	1.216	0.841	1.011

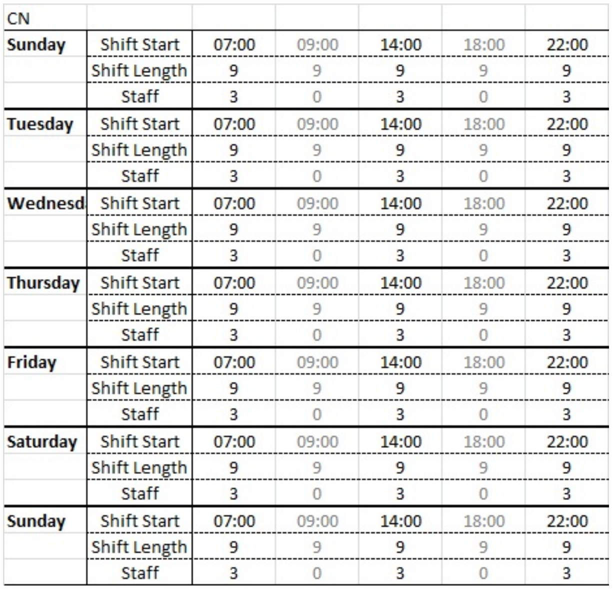


CM						
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	3	0	2
Tuesday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	3	0	2
Wednesd	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	3	0	2
Thursday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	3	0	2
Friday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	3	0	2
Saturday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	3	0	2
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	3	0	2

Figure A-4: Trial problem based upon the CM region of Leicestershire.

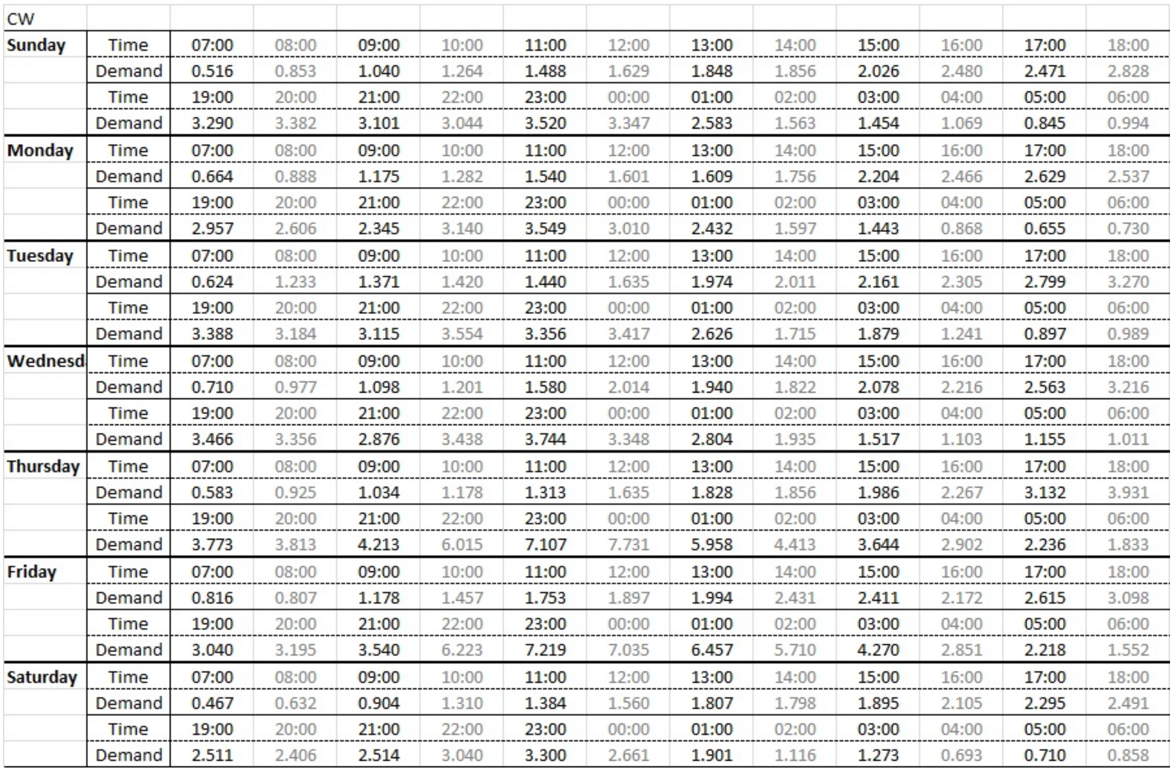


CN													
Sunday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.599	0.991	1.537	1.661	2.123	2.184	2.227	2.368	2.638	3.149	3.520	3.236
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.460	3.543	3.598	4.758	4.706	3.968	2.483	2.227	2.195	1.322	0.776	1.218
Monday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.807	1.152	1.569	1.736	1.744	2.149	2.118	2.290	2.759	2.917	3.086	3.244
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	2.813	2.871	3.014	3.864	4.080	3.566	2.903	2.239	1.908	1.511	0.960	0.931
Tuesday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.632	1.207	1.543	1.655	1.917	2.241	2.305	2.336	2.483	2.750	3.026	3.049
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	2.856	3.034	3.080	3.789	4.346	3.356	2.586	2.028	1.839	1.115	1.109	0.856
Wednesd	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.724	1.126	1.776	1.943	2.270	2.480	2.569	2.761	3.207	3.704	3.713	3.796
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.698	3.922	4.026	5.678	5.729	5.468	4.253	2.862	2.937	2.379	2.029	1.408
Thursday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.624	0.934	1.351	1.695	1.879	2.124	2.270	2.566	2.687	2.876	2.997	2.879
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	2.983	3.448	3.471	5.026	6.141	5.824	4.284	3.467	3.241	2.391	1.885	1.891
Friday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.750	0.825	1.138	1.422	1.621	1.764	1.822	2.138	2.256	2.457	2.644	3.060
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.193	3.399	3.080	4.666	6.134	6.124	4.738	4.395	3.598	2.759	2.333	1.678
Saturday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.753	0.677	0.878	1.267	1.654	1.642	1.886	2.176	2.580	2.983	3.270	3.699
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.824	3.653	3.878	5.526	5.472	4.772	4.038	2.910	2.614	1.710	0.920	0.989

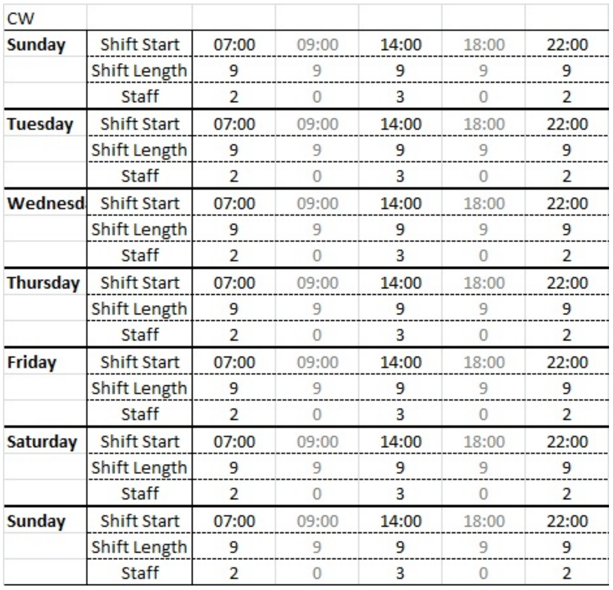


CN						
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	3	0	3	0	3
Tuesday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	3	0	3	0	3
Wednesd	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	3	0	3	0	3
Thursday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	3	0	3	0	3
Friday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	3	0	3	0	3
Saturday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	3	0	3	0	3
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	3	0	3	0	3

Figure A-5: Trial problem based upon the CN region of Leicestershire.

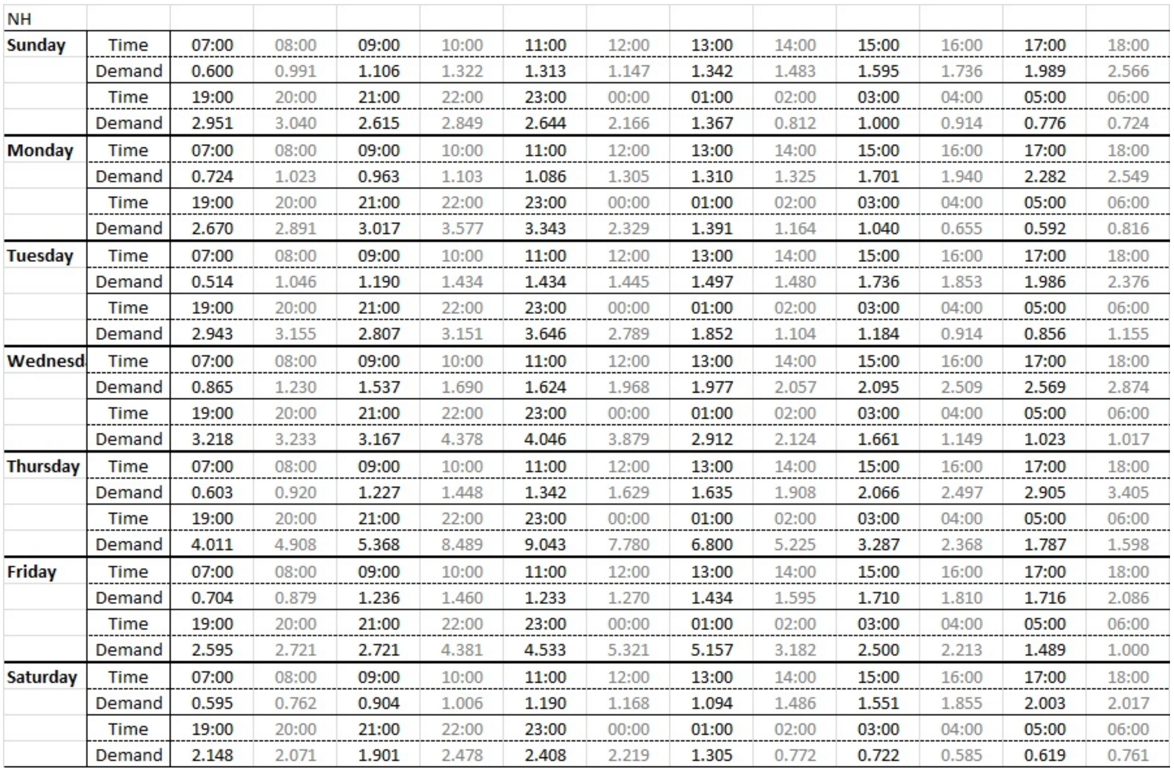


CW													
Sunday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.516	0.853	1.040	1.264	1.488	1.629	1.848	1.856	2.026	2.480	2.471	2.828
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.290	3.382	3.101	3.044	3.520	3.347	2.583	1.563	1.454	1.069	0.845	0.994
Monday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.664	0.888	1.175	1.282	1.540	1.601	1.609	1.756	2.204	2.466	2.629	2.537
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	2.957	2.606	2.345	3.140	3.549	3.010	2.432	1.597	1.443	0.868	0.655	0.730
Tuesday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.624	1.233	1.371	1.420	1.440	1.635	1.974	2.011	2.161	2.305	2.799	3.270
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.388	3.184	3.115	3.554	3.356	3.417	2.626	1.715	1.879	1.241	0.897	0.989
Wednesd	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.710	0.977	1.098	1.201	1.580	2.014	1.940	1.822	2.078	2.216	2.563	3.216
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.466	3.356	2.876	3.438	3.744	3.348	2.804	1.935	1.517	1.103	1.155	1.011
Thursday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.583	0.925	1.034	1.178	1.313	1.635	1.828	1.856	1.986	2.267	3.132	3.931
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.773	3.813	4.213	6.015	7.107	7.731	5.958	4.413	3.644	2.902	2.236	1.833
Friday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.816	0.807	1.178	1.457	1.753	1.897	1.994	2.431	2.411	2.172	2.615	3.098
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.040	3.195	3.540	6.223	7.219	7.035	6.457	5.710	4.270	2.851	2.218	1.552
Saturday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.467	0.632	0.904	1.310	1.384	1.560	1.807	1.798	1.895	2.105	2.295	2.491
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	2.511	2.406	2.514	3.040	3.300	2.661	1.901	1.116	1.273	0.693	0.710	0.858

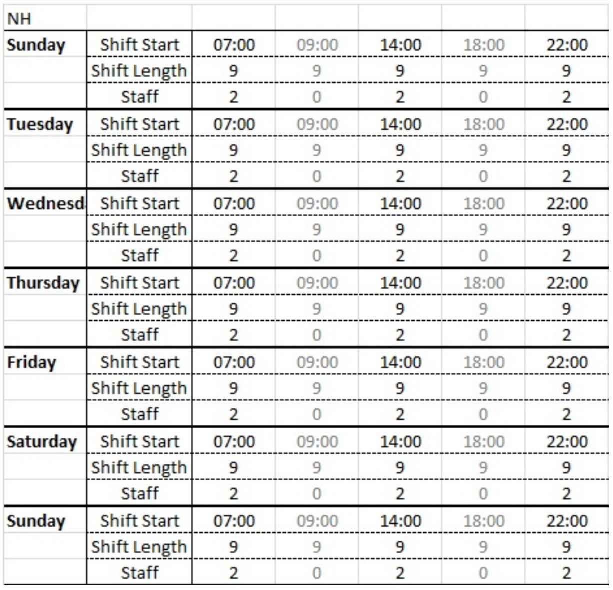


CW						
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	3	0	2
Tuesday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	3	0	2
Wednesd	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	3	0	2
Thursday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	3	0	2
Friday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	3	0	2
Saturday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	3	0	2
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	3	0	2

Figure A-6: Trial problem based upon the CW region of Leicestershire.

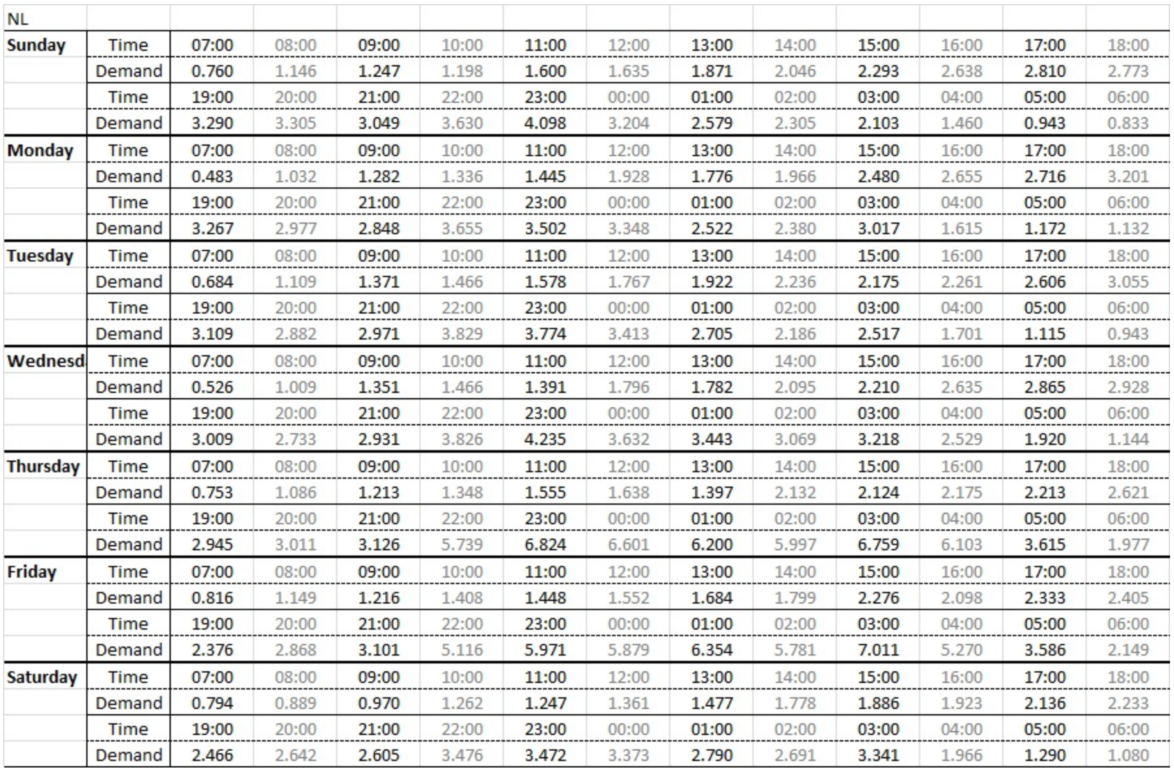


NH													
Sunday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.600	0.991	1.106	1.322	1.313	1.147	1.342	1.483	1.595	1.736	1.989	2.566
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	2.951	3.040	2.615	2.849	2.644	2.166	1.367	0.812	1.000	0.914	0.776	0.724
Monday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.724	1.023	0.963	1.103	1.086	1.305	1.310	1.325	1.701	1.940	2.282	2.549
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	2.670	2.891	3.017	3.577	3.343	2.329	1.391	1.164	1.040	0.655	0.592	0.816
Tuesday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.514	1.046	1.190	1.434	1.434	1.445	1.497	1.480	1.736	1.853	1.986	2.376
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	2.943	3.155	2.807	3.151	3.646	2.789	1.852	1.104	1.184	0.914	0.856	1.155
Wednesd	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.865	1.230	1.537	1.690	1.624	1.968	1.977	2.057	2.095	2.509	2.569	2.874
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.218	3.233	3.167	4.378	4.046	3.879	2.912	2.124	1.661	1.149	1.023	1.017
Thursday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.603	0.920	1.227	1.448	1.342	1.629	1.635	1.908	2.066	2.497	2.905	3.405
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	4.011	4.908	5.368	8.489	9.043	7.780	6.800	5.225	3.287	2.368	1.787	1.598
Friday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.704	0.879	1.236	1.460	1.233	1.270	1.434	1.595	1.710	1.810	1.716	2.086
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	2.595	2.721	2.721	4.381	4.533	5.321	5.157	3.182	2.500	2.213	1.489	1.000
Saturday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.595	0.762	0.904	1.006	1.190	1.168	1.094	1.486	1.551	1.855	2.003	2.017
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	2.148	2.071	1.901	2.478	2.408	2.219	1.305	0.772	0.722	0.585	0.619	0.761

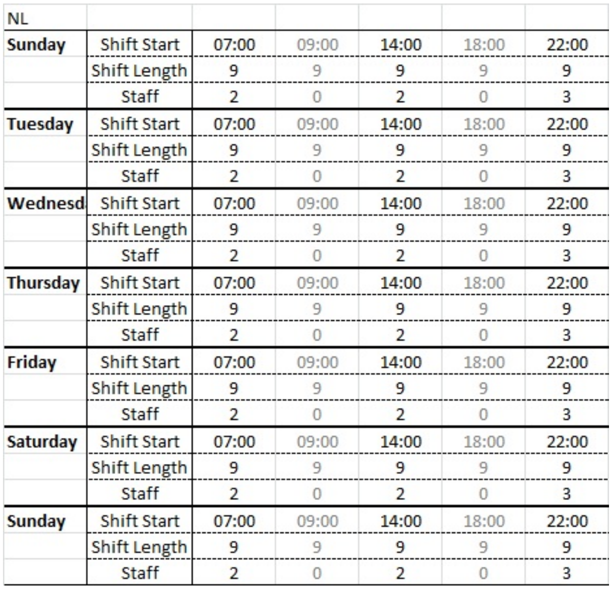


NH						
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2
Tuesday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2
Wednesd	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2
Thursday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2
Friday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2
Saturday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2

Figure A-7: Trial problem based upon the NH region of Leicestershire.

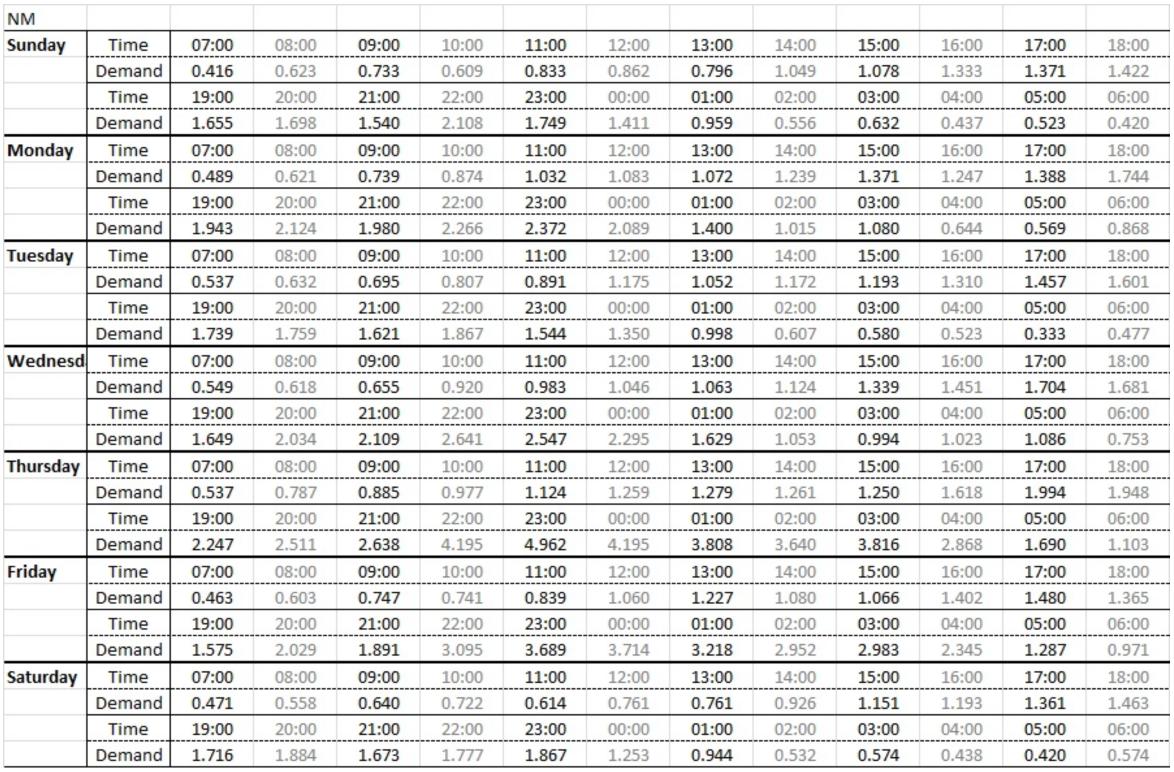


NL													
Sunday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.760	1.146	1.247	1.198	1.600	1.635	1.871	2.046	2.293	2.638	2.810	2.773
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.290	3.305	3.049	3.630	4.098	3.204	2.579	2.305	2.103	1.460	0.943	0.833
Monday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.483	1.032	1.282	1.336	1.445	1.928	1.776	1.966	2.480	2.655	2.716	3.201
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.267	2.977	2.848	3.655	3.502	3.348	2.522	2.380	3.017	1.615	1.172	1.132
Tuesday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.684	1.109	1.371	1.466	1.578	1.767	1.922	2.236	2.175	2.261	2.606	3.055
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.109	2.882	2.971	3.829	3.774	3.413	2.705	2.186	2.517	1.701	1.115	0.943
Wednesd	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.526	1.009	1.351	1.466	1.391	1.796	1.782	2.095	2.210	2.635	2.865	2.928
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.009	2.733	2.931	3.826	4.235	3.632	3.443	3.069	3.218	2.529	1.920	1.144
Thursday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.753	1.086	1.213	1.348	1.555	1.638	1.397	2.132	2.124	2.175	2.213	2.621
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	2.945	3.011	3.126	5.739	6.824	6.601	6.200	5.997	6.759	6.103	3.615	1.977
Friday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.816	1.149	1.216	1.408	1.448	1.552	1.684	1.799	2.276	2.098	2.333	2.405
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	2.376	2.868	3.101	5.116	5.971	5.879	6.354	5.781	7.011	5.270	3.586	2.149
Saturday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.794	0.889	0.970	1.262	1.247	1.361	1.477	1.778	1.886	1.923	2.136	2.233
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	2.466	2.642	2.605	3.476	3.472	3.373	2.790	2.691	3.341	1.966	1.290	1.080

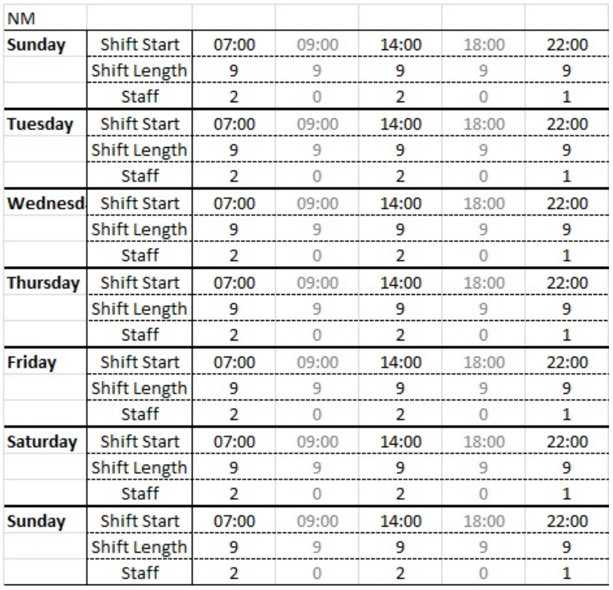


NL						
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	3
Tuesday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	3
Wednesd	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	3
Thursday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	3
Friday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	3
Saturday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	3
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	3

Figure A-8: Trial problem based upon the NL region of Leicestershire.



NM													
Sunday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.416	0.623	0.733	0.609	0.833	0.862	0.796	1.049	1.078	1.333	1.371	1.422
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	1.655	1.698	1.540	2.108	1.749	1.411	0.959	0.556	0.632	0.437	0.523	0.420
Monday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.489	0.621	0.739	0.874	1.032	1.083	1.072	1.239	1.371	1.247	1.388	1.744
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	1.943	2.124	1.980	2.266	2.372	2.089	1.400	1.015	1.080	0.644	0.569	0.868
Tuesday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.537	0.632	0.695	0.807	0.891	1.175	1.052	1.172	1.193	1.310	1.457	1.601
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	1.739	1.759	1.621	1.867	1.544	1.350	0.998	0.607	0.580	0.523	0.333	0.477
Wednesd	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.549	0.618	0.655	0.920	0.983	1.046	1.063	1.124	1.339	1.451	1.704	1.681
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	1.649	2.034	2.109	2.641	2.547	2.295	1.629	1.053	0.994	1.023	1.086	0.753
Thursday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.537	0.787	0.885	0.977	1.124	1.259	1.279	1.261	1.250	1.618	1.994	1.948
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	2.247	2.511	2.638	4.195	4.962	4.195	3.808	3.640	3.816	2.868	1.690	1.103
Friday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.463	0.603	0.747	0.741	0.839	1.060	1.227	1.080	1.066	1.402	1.480	1.365
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	1.575	2.029	1.891	3.095	3.689	3.714	3.218	2.952	2.983	2.345	1.287	0.971
Saturday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.471	0.558	0.640	0.722	0.614	0.761	0.761	0.926	1.151	1.193	1.361	1.463
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	1.716	1.884	1.673	1.777	1.867	1.253	0.944	0.532	0.574	0.438	0.420	0.574



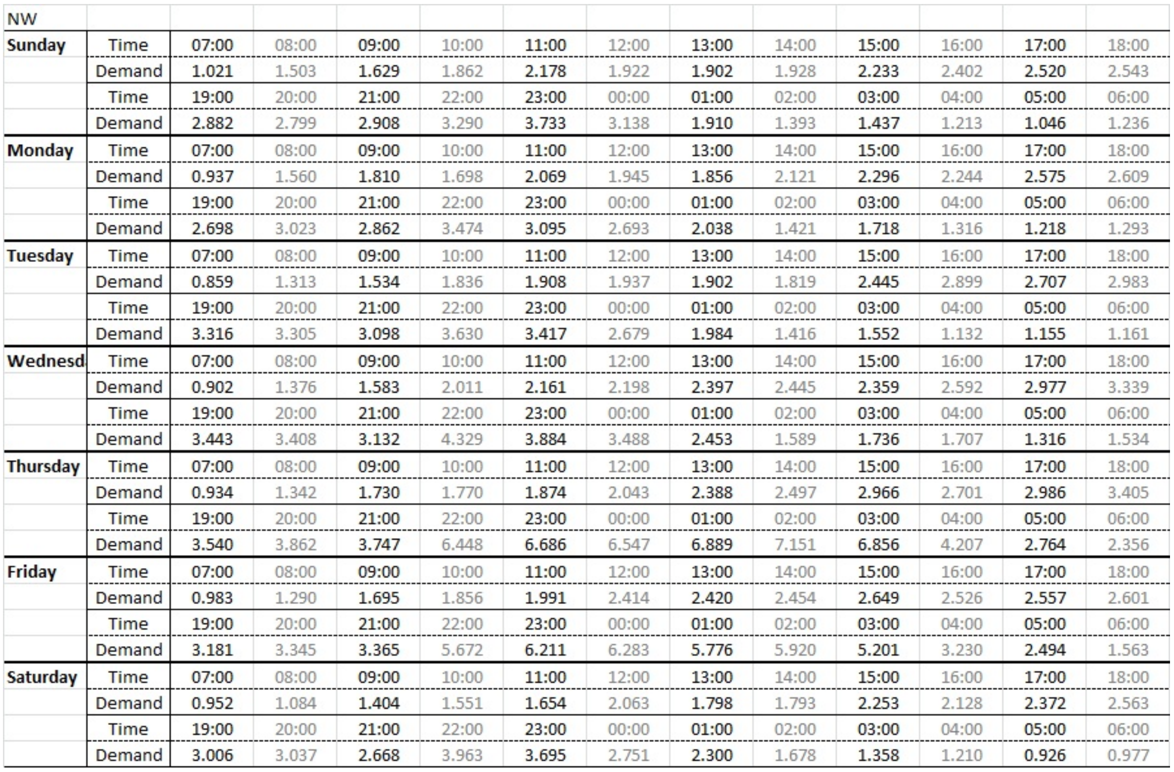
NM						
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	1
Tuesday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	1
Wednesd	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	1
Thursday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	1
Friday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	1
Saturday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	1
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	1

Figure A-9: Trial problem based upon the NM region of Leicestershire.

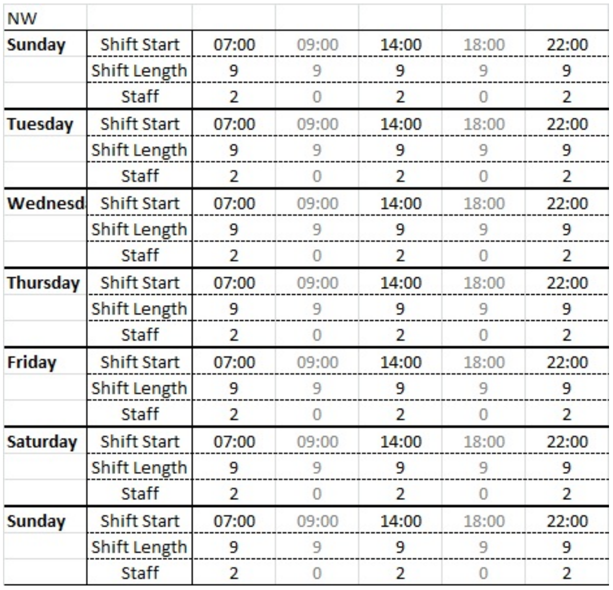
NR													
Sunday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.550	0.773	0.810	0.747	0.583	0.767	0.756	0.724	0.693	0.744	0.807	0.925
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	0.874	0.807	0.713	0.937	0.881	0.760	0.621	0.534	0.477	0.454	0.213	0.489
Monday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.572	0.776	0.716	0.773	0.713	0.756	0.701	0.773	0.816	0.931	1.080	1.115
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	0.989	0.796	0.856	1.196	1.015	0.771	0.775	0.514	0.644	0.477	0.408	0.425
Tuesday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.411	0.618	0.601	0.629	0.592	0.710	0.810	0.859	0.925	0.813	0.908	0.934
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	0.945	0.925	0.736	0.997	1.047	0.990	0.541	0.396	0.575	0.299	0.316	0.402
Wednesd	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.351	0.388	0.606	0.661	0.756	0.687	0.750	0.704	0.761	0.874	0.954	0.977
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	0.920	0.851	0.675	1.006	1.008	0.918	0.842	0.617	0.598	0.489	0.443	0.391
Thursday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.376	0.486	0.635	0.578	0.629	0.830	0.635	0.713	0.845	0.862	0.971	1.023
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	0.807	1.138	1.144	1.824	2.144	2.407	2.095	1.570	1.506	0.971	0.408	0.414
Friday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.348	0.437	0.624	0.667	0.678	0.905	0.793	0.819	0.977	0.954	0.802	0.825
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	1.034	0.937	1.029	2.388	2.763	2.517	2.394	2.149	1.707	1.195	1.006	0.707
Saturday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.330	0.421	0.509	0.423	0.563	0.707	0.616	0.673	0.710	0.778	0.719	0.773
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	0.881	0.892	0.665	0.907	1.000	0.695	0.476	0.442	0.580	0.443	0.307	0.545

NR						
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	1	0	1	0	1
Tuesday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	1	0	1	0	1
Wednesd	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	1	0	1	0	1
Thursday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	1	0	1	0	1
Friday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	1	0	1	0	1
Saturday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	1	0	1	0	1
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	1	0	1	0	1

Figure A-10: Trial problem based upon the NR region of Leicestershire.

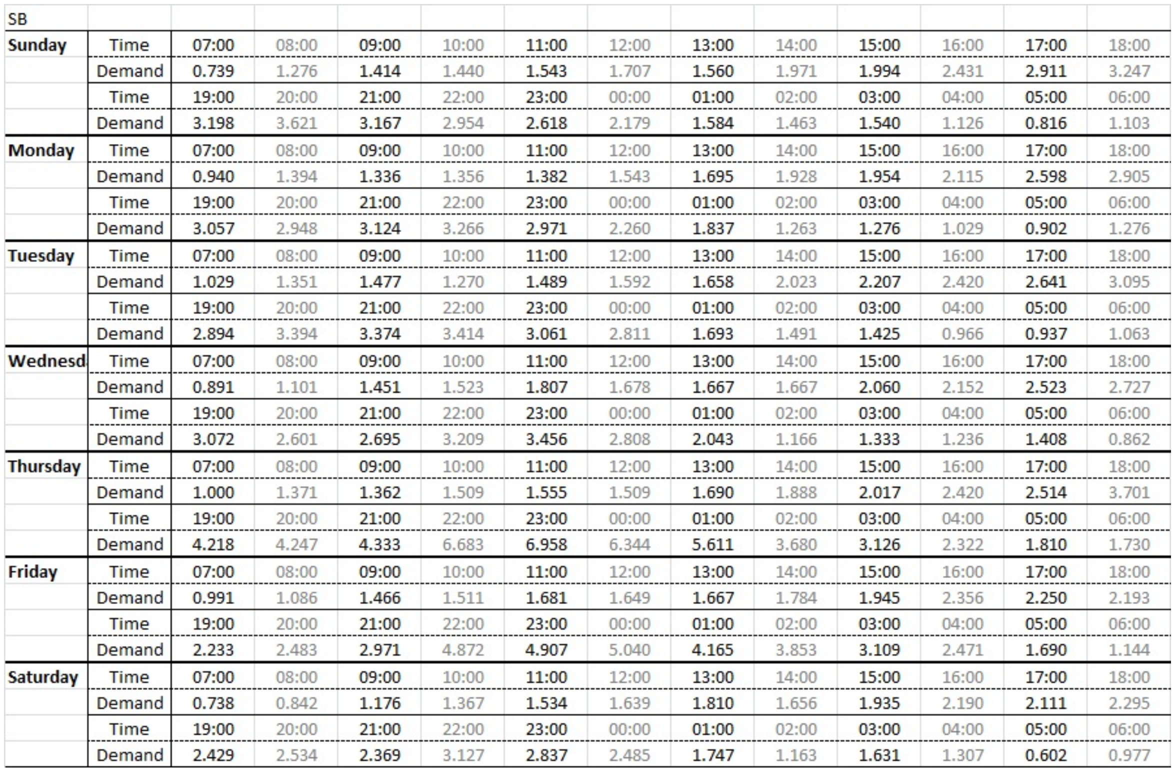


NW													
Sunday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	1.021	1.503	1.629	1.862	2.178	1.922	1.902	1.928	2.233	2.402	2.520	2.543
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	2.882	2.799	2.908	3.290	3.733	3.138	1.910	1.393	1.437	1.213	1.046	1.236
Monday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.937	1.560	1.810	1.698	2.069	1.945	1.856	2.121	2.296	2.244	2.575	2.609
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	2.698	3.023	2.862	3.474	3.095	2.693	2.038	1.421	1.718	1.316	1.218	1.293
Tuesday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.859	1.313	1.534	1.836	1.908	1.937	1.902	1.819	2.445	2.899	2.707	2.983
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.316	3.305	3.098	3.630	3.417	2.679	1.984	1.416	1.552	1.132	1.155	1.161
Wednesd	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.902	1.376	1.583	2.011	2.161	2.198	2.397	2.445	2.359	2.592	2.977	3.339
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.443	3.408	3.132	4.329	3.884	3.488	2.453	1.589	1.736	1.707	1.316	1.534
Thursday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.934	1.342	1.730	1.770	1.874	2.043	2.388	2.497	2.966	2.701	2.986	3.405
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.540	3.862	3.747	6.448	6.686	6.547	6.889	7.151	6.856	4.207	2.764	2.356
Friday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.983	1.290	1.695	1.856	1.991	2.414	2.420	2.454	2.649	2.526	2.557	2.601
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.181	3.345	3.365	5.672	6.211	6.283	5.776	5.920	5.201	3.230	2.494	1.563
Saturday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.952	1.084	1.404	1.551	1.654	2.063	1.798	1.793	2.253	2.128	2.372	2.563
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.006	3.037	2.668	3.963	3.695	2.751	2.300	1.678	1.358	1.210	0.926	0.977

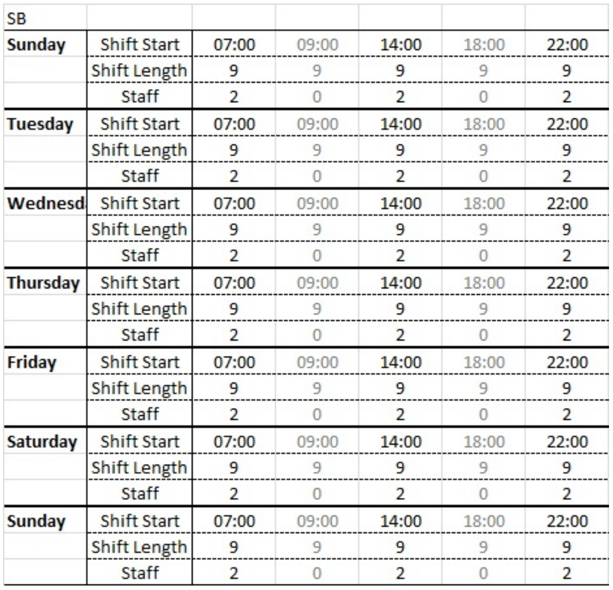


NW						
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2
Tuesday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2
Wednesd	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2
Thursday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2
Friday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2
Saturday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2

Figure A-11: Trial problem based upon the NW region of Leicestershire.

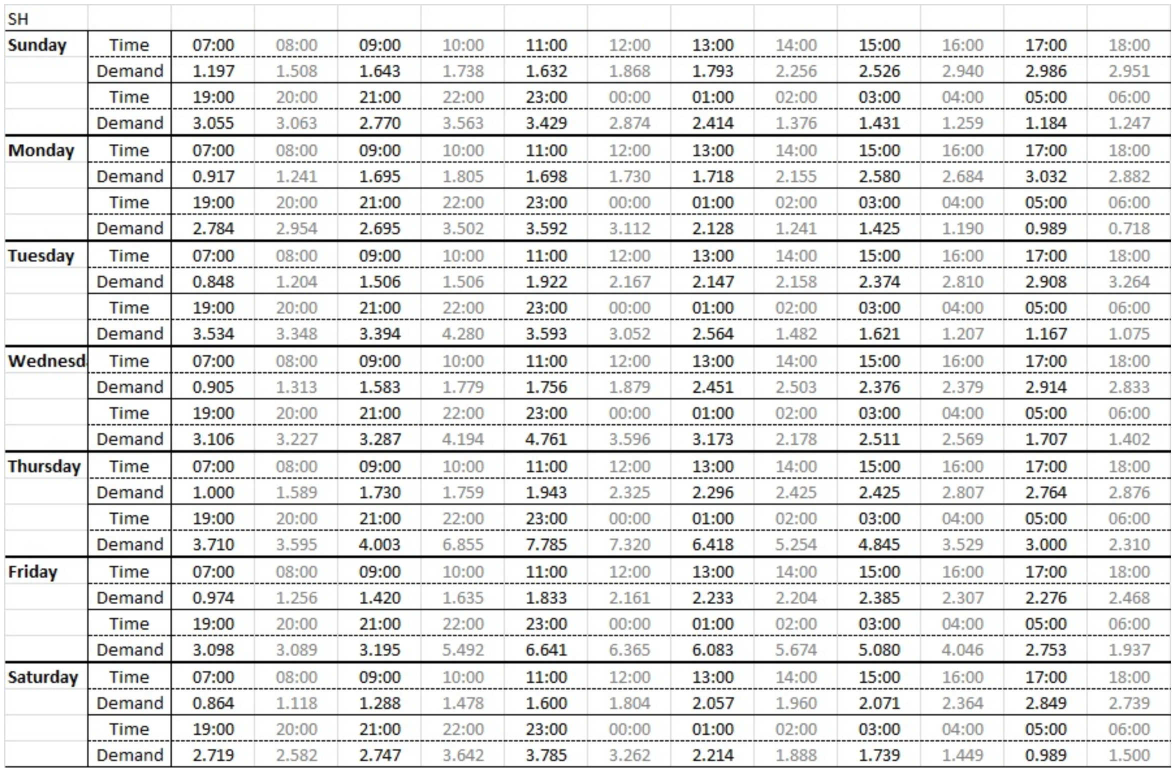


SB												
Sunday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.739	1.276	1.414	1.440	1.543	1.707	1.560	1.971	1.994	2.431	2.911	3.247
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.198	3.621	3.167	2.954	2.618	2.179	1.584	1.463	1.540	1.126	0.816	1.103
Monday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.940	1.394	1.336	1.356	1.382	1.543	1.695	1.928	1.954	2.115	2.598	2.905
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.057	2.948	3.124	3.266	2.971	2.260	1.837	1.263	1.276	1.029	0.902	1.276
Tuesday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	1.029	1.351	1.477	1.270	1.489	1.592	1.658	2.023	2.207	2.420	2.641	3.095
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	2.894	3.394	3.374	3.414	3.061	2.811	1.693	1.491	1.425	0.966	0.937	1.063
Wednesd	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.891	1.101	1.451	1.523	1.807	1.678	1.667	1.667	2.060	2.152	2.523	2.727
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.072	2.601	2.695	3.209	3.456	2.808	2.043	1.166	1.333	1.236	1.408	0.862
Thursday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	1.000	1.371	1.362	1.509	1.555	1.509	1.690	1.888	2.017	2.420	2.514	3.701
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	4.218	4.247	4.333	6.683	6.958	6.344	5.611	3.680	3.126	2.322	1.810	1.730
Friday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.991	1.086	1.466	1.511	1.681	1.649	1.667	1.784	1.945	2.356	2.250	2.193
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	2.233	2.483	2.971	4.872	4.907	5.040	4.165	3.853	3.109	2.471	1.690	1.144
Saturday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.738	0.842	1.176	1.367	1.534	1.639	1.810	1.656	1.935	2.190	2.111	2.295
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	2.429	2.534	2.369	3.127	2.837	2.485	1.747	1.163	1.631	1.307	0.602	0.977

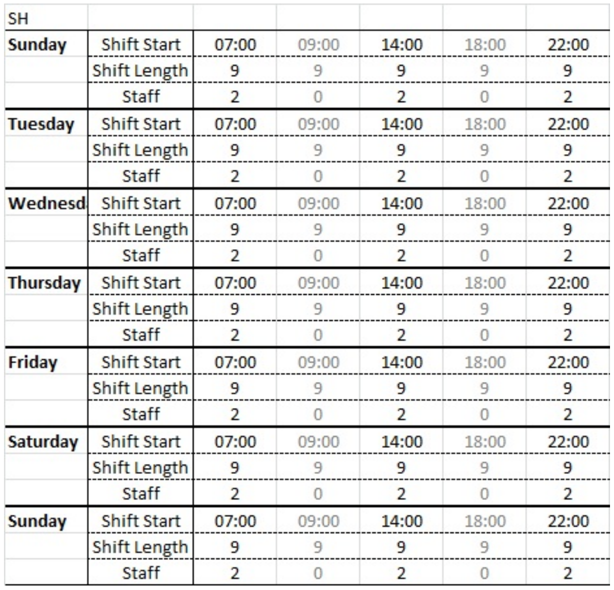


SB						
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2
Tuesday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2
Wednesd	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2
Thursday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2
Friday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2
Saturday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2

Figure A-12: Trial problem based upon the SB region of Leicestershire.

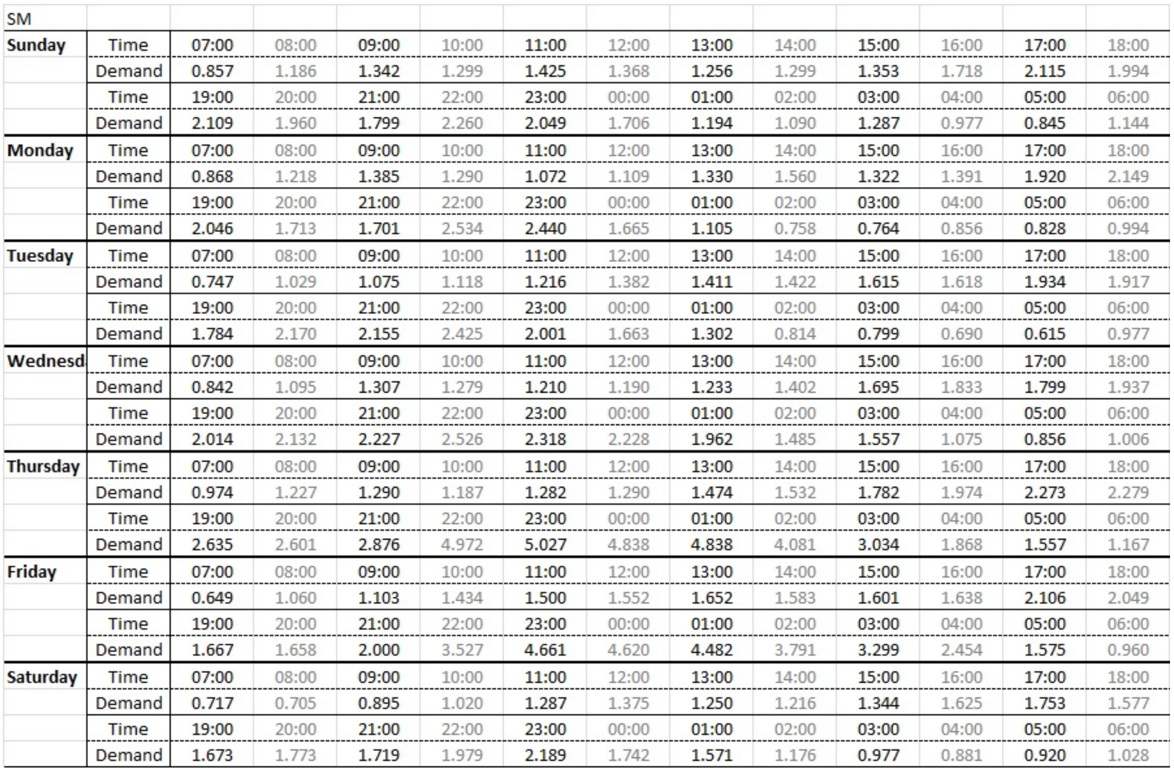


SH													
Sunday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	1.197	1.508	1.643	1.738	1.632	1.868	1.793	2.256	2.526	2.940	2.986	2.951
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.055	3.063	2.770	3.563	3.429	2.874	2.414	1.376	1.431	1.259	1.184	1.247
Monday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.917	1.241	1.695	1.805	1.698	1.730	1.718	2.155	2.580	2.684	3.032	2.882
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	2.784	2.954	2.695	3.502	3.592	3.112	2.128	1.241	1.425	1.190	0.989	0.718
Tuesday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.848	1.204	1.506	1.506	1.922	2.167	2.147	2.158	2.374	2.810	2.908	3.264
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.534	3.348	3.394	4.280	3.593	3.052	2.564	1.482	1.621	1.207	1.167	1.075
Wednesd	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.905	1.313	1.583	1.779	1.756	1.879	2.451	2.503	2.376	2.379	2.914	2.833
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.106	3.227	3.287	4.194	4.761	3.596	3.173	2.178	2.511	2.569	1.707	1.402
Thursday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	1.000	1.589	1.730	1.759	1.943	2.325	2.296	2.425	2.425	2.807	2.764	2.876
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.710	3.595	4.003	6.855	7.785	7.320	6.418	5.254	4.845	3.529	3.000	2.310
Friday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.974	1.256	1.420	1.635	1.833	2.161	2.233	2.204	2.385	2.307	2.276	2.468
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	3.098	3.089	3.195	5.492	6.641	6.365	6.083	5.674	5.080	4.046	2.753	1.937
Saturday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.864	1.118	1.288	1.478	1.600	1.804	2.057	1.960	2.071	2.364	2.849	2.739
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	2.719	2.582	2.747	3.642	3.785	3.262	2.214	1.888	1.739	1.449	0.989	1.500

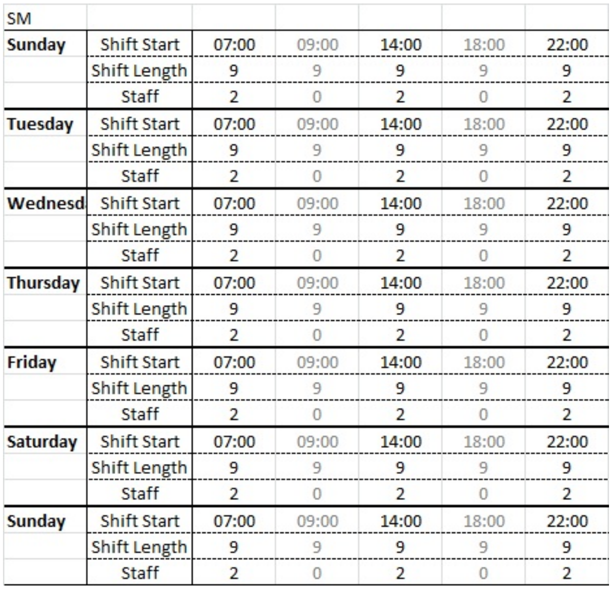


SH						
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2
Tuesday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2
Wednesd	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2
Thursday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2
Friday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2
Saturday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2

Figure A-13: Trial problem based upon the SH region of Leicestershire.

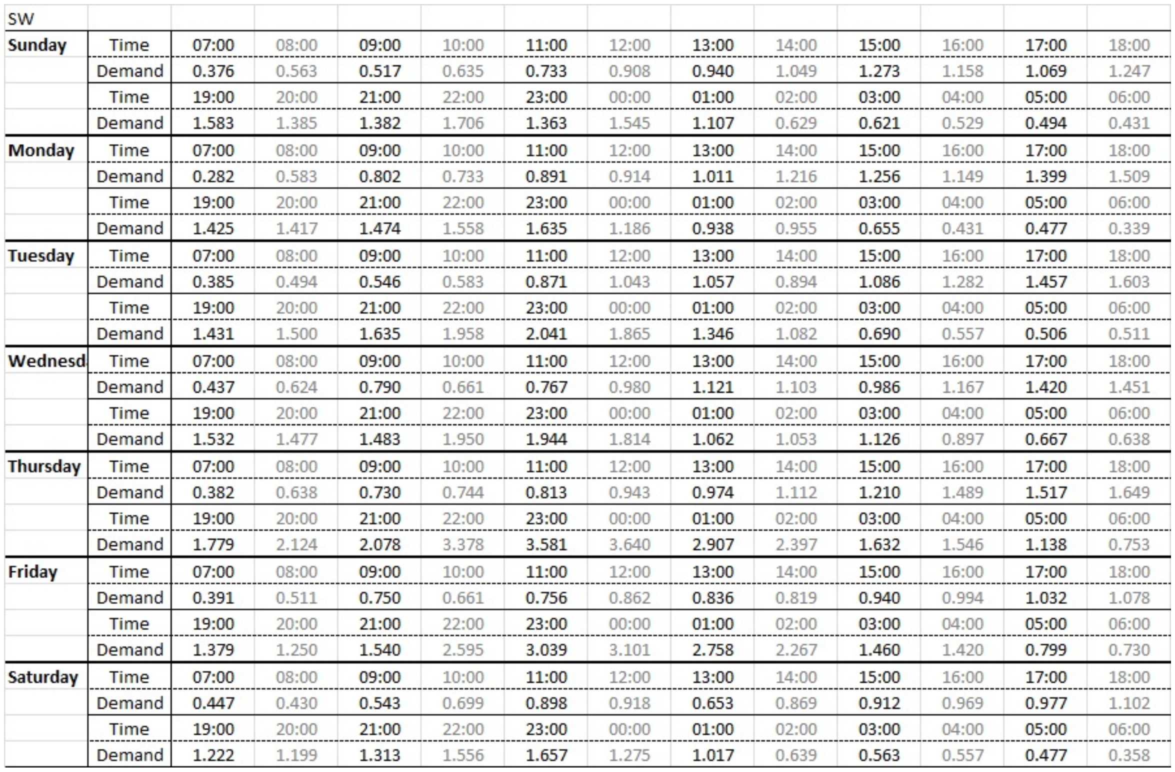


SM													
Sunday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.857	1.186	1.342	1.299	1.425	1.368	1.256	1.299	1.353	1.718	2.115	1.994
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	2.109	1.960	1.799	2.260	2.049	1.706	1.194	1.090	1.287	0.977	0.845	1.144
Monday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.868	1.218	1.385	1.290	1.072	1.109	1.330	1.560	1.322	1.391	1.920	2.149
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	2.046	1.713	1.701	2.534	2.440	1.665	1.105	0.758	0.764	0.856	0.828	0.994
Tuesday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.747	1.029	1.075	1.118	1.216	1.382	1.411	1.422	1.615	1.618	1.934	1.917
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	1.784	2.170	2.155	2.425	2.001	1.663	1.302	0.814	0.799	0.690	0.615	0.977
Wednesd	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.842	1.095	1.307	1.279	1.210	1.190	1.233	1.402	1.695	1.833	1.799	1.937
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	2.014	2.132	2.227	2.526	2.318	2.228	1.962	1.485	1.557	1.075	0.856	1.006
Thursday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.974	1.227	1.290	1.187	1.282	1.290	1.474	1.532	1.782	1.974	2.273	2.279
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	2.635	2.601	2.876	4.972	5.027	4.838	4.838	4.081	3.034	1.868	1.557	1.167
Friday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.649	1.060	1.103	1.434	1.500	1.552	1.652	1.583	1.601	1.638	2.106	2.049
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	1.667	1.658	2.000	3.527	4.661	4.620	4.482	3.791	3.299	2.454	1.575	0.960
Saturday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.717	0.705	0.895	1.020	1.287	1.375	1.250	1.216	1.344	1.625	1.753	1.577
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	1.673	1.773	1.719	1.979	2.189	1.742	1.571	1.176	0.977	0.881	0.920	1.028

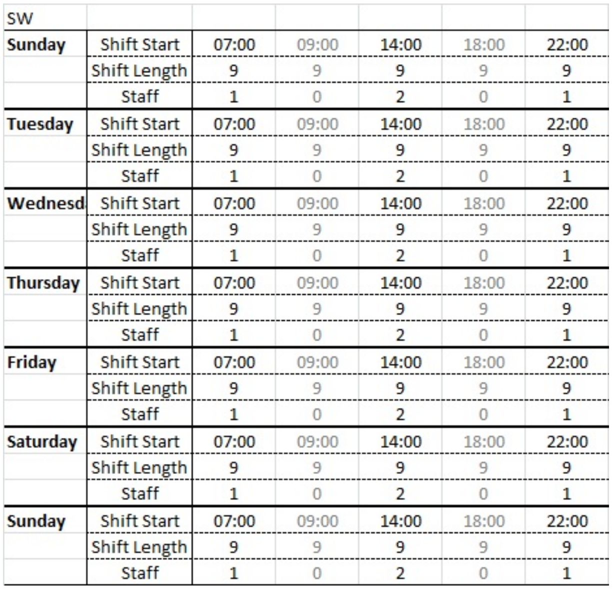


SM						
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2
Tuesday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2
Wednesd	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2
Thursday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2
Friday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2
Saturday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	2	0	2	0	2

Figure A-14: Trial problem based upon the SM region of Leicestershire.

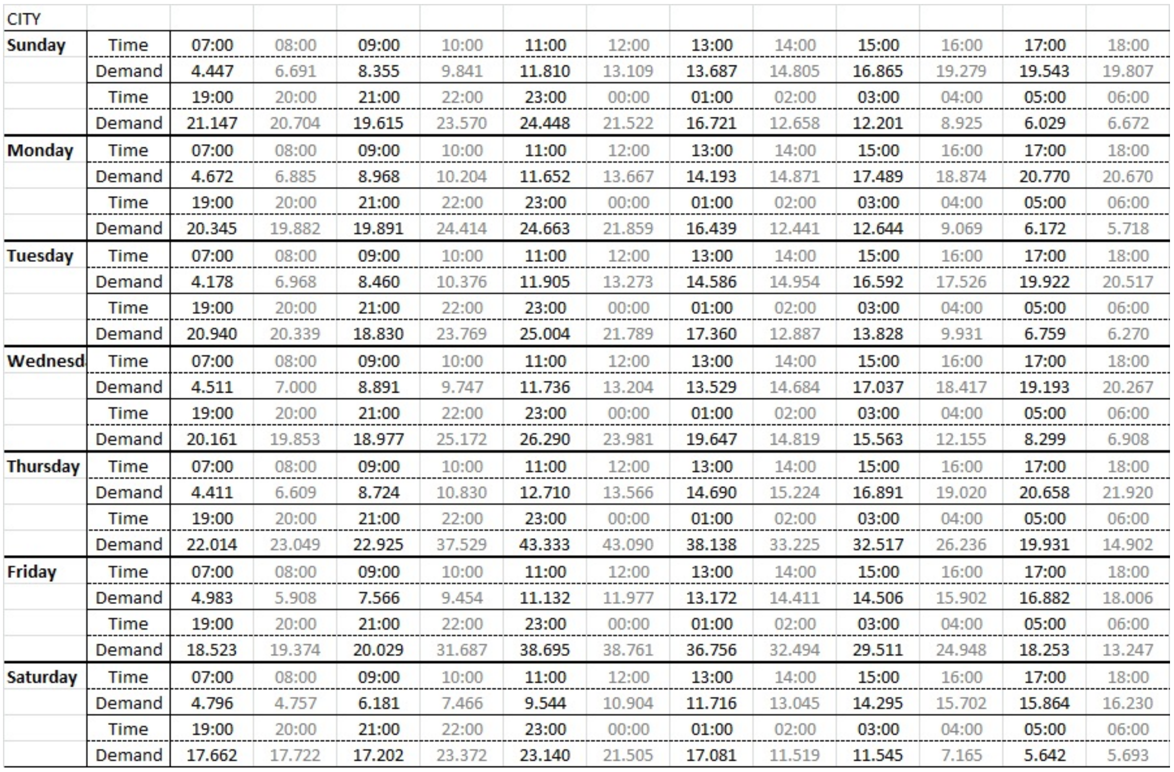


SW													
Sunday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.376	0.563	0.517	0.635	0.733	0.908	0.940	1.049	1.273	1.158	1.069	1.247
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	1.583	1.385	1.382	1.706	1.363	1.545	1.107	0.629	0.621	0.529	0.494	0.431
Monday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.282	0.583	0.802	0.733	0.891	0.914	1.011	1.216	1.256	1.149	1.399	1.509
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	1.425	1.417	1.474	1.558	1.635	1.186	0.938	0.955	0.655	0.431	0.477	0.339
Tuesday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.385	0.494	0.546	0.583	0.871	1.043	1.057	0.894	1.086	1.282	1.457	1.603
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	1.431	1.500	1.635	1.958	2.041	1.865	1.346	1.082	0.690	0.557	0.506	0.511
Wednesd	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.437	0.624	0.790	0.661	0.767	0.980	1.121	1.103	0.986	1.167	1.420	1.451
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	1.532	1.477	1.483	1.950	1.944	1.814	1.062	1.053	1.126	0.897	0.667	0.638
Thursday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.382	0.638	0.730	0.744	0.813	0.943	0.974	1.112	1.210	1.489	1.517	1.649
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	1.779	2.124	2.078	3.378	3.581	3.640	2.907	2.397	1.632	1.546	1.138	0.753
Friday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.391	0.511	0.750	0.661	0.756	0.862	0.836	0.819	0.940	0.994	1.032	1.078
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	1.379	1.250	1.540	2.595	3.039	3.101	2.758	2.267	1.460	1.420	0.799	0.730
Saturday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	0.447	0.430	0.543	0.699	0.898	0.918	0.653	0.869	0.912	0.969	0.977	1.102
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	1.222	1.199	1.313	1.556	1.657	1.275	1.017	0.639	0.563	0.557	0.477	0.358

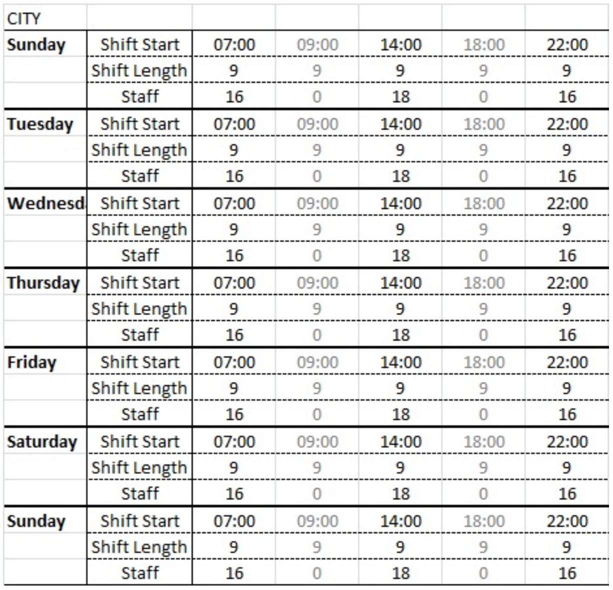


SW						
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	1	0	2	0	1
Tuesday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	1	0	2	0	1
Wednesd	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	1	0	2	0	1
Thursday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	1	0	2	0	1
Friday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	1	0	2	0	1
Saturday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	1	0	2	0	1
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	1	0	2	0	1

Figure A-15: Trial problem based upon the SW region of Leicestershire.

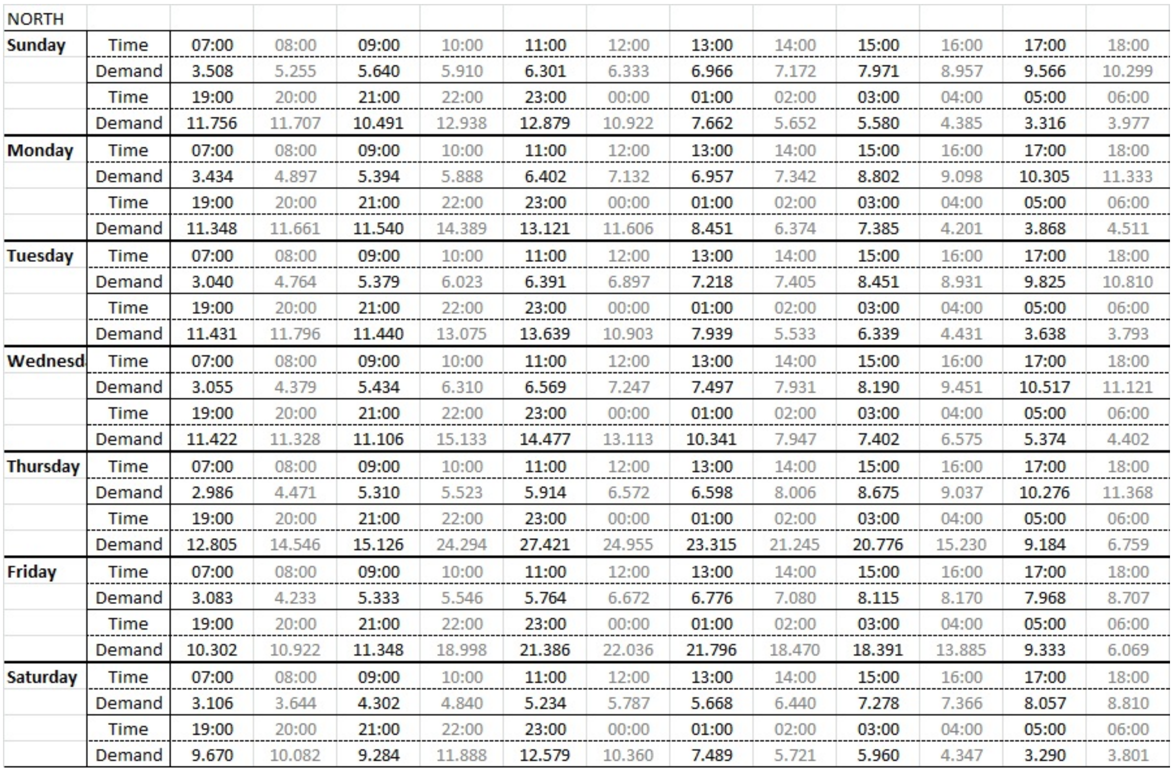


CITY													
Sunday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	4.447	6.691	8.355	9.841	11.810	13.109	13.687	14.805	16.865	19.279	19.543	19.807
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	21.147	20.704	19.615	23.570	24.448	21.522	16.721	12.658	12.201	8.925	6.029	6.672
Monday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	4.672	6.885	8.968	10.204	11.652	13.667	14.193	14.871	17.489	18.874	20.770	20.670
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	20.345	19.882	19.891	24.414	24.663	21.859	16.439	12.441	12.644	9.069	6.172	5.718
Tuesday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	4.178	6.968	8.460	10.376	11.905	13.273	14.586	14.954	16.592	17.526	19.922	20.517
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	20.940	20.339	18.830	23.769	25.004	21.789	17.360	12.887	13.828	9.931	6.759	6.270
Wednesd	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	4.511	7.000	8.891	9.747	11.736	13.204	13.529	14.684	17.037	18.417	19.193	20.267
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	20.161	19.853	18.977	25.172	26.290	23.981	19.647	14.819	15.563	12.155	8.299	6.908
Thursday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	4.411	6.609	8.724	10.830	12.710	13.566	14.690	15.224	16.891	19.020	20.658	21.920
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	22.014	23.049	22.925	37.529	43.333	43.090	38.138	33.225	32.517	26.236	19.931	14.902
Friday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	4.983	5.908	7.566	9.454	11.132	11.977	13.172	14.411	14.506	15.902	16.882	18.006
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	18.523	19.374	20.029	31.687	38.695	38.761	36.756	32.494	29.511	24.948	18.253	13.247
Saturday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	4.796	4.757	6.181	7.466	9.544	10.904	11.716	13.045	14.295	15.702	15.864	16.230
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	17.662	17.722	17.202	23.372	23.140	21.505	17.081	11.519	11.545	7.165	5.642	5.693

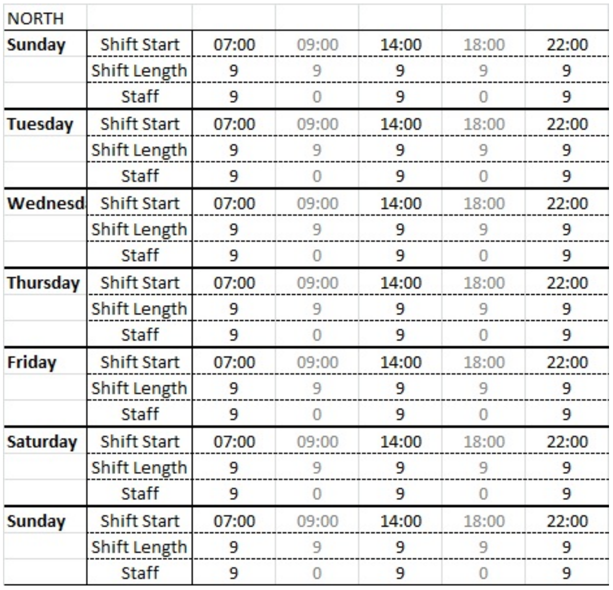


CITY						
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	16	0	18	0	16
Tuesday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	16	0	18	0	16
Wednesd	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	16	0	18	0	16
Thursday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	16	0	18	0	16
Friday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	16	0	18	0	16
Saturday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	16	0	18	0	16
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	16	0	18	0	16

Figure A-16: Trial problem based upon the City region of Leicestershire.

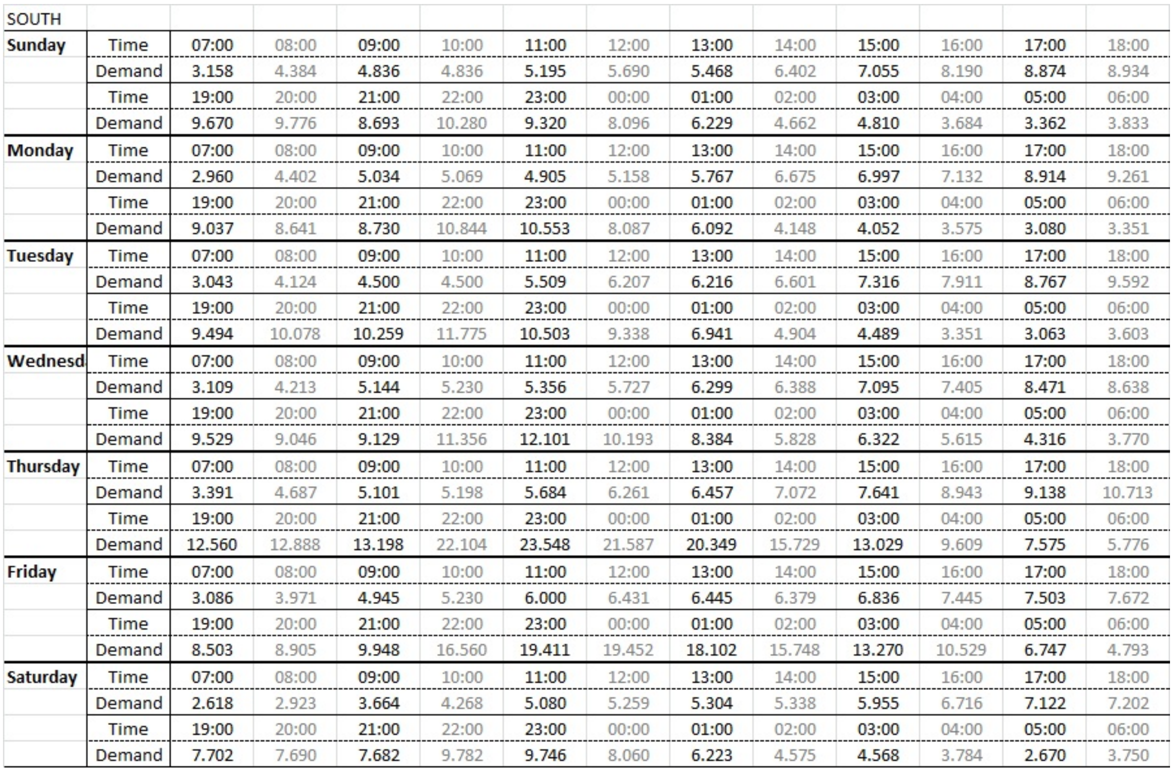


NORTH													
Sunday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	3.508	5.255	5.640	5.910	6.301	6.333	6.966	7.172	7.971	8.957	9.566	10.299
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	11.756	11.707	10.491	12.938	12.879	10.922	7.662	5.652	5.580	4.385	3.316	3.977
Monday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	3.434	4.897	5.394	5.888	6.402	7.132	6.957	7.342	8.802	9.098	10.305	11.333
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	11.348	11.661	11.540	14.389	13.121	11.606	8.451	6.374	7.385	4.201	3.868	4.511
Tuesday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	3.040	4.764	5.379	6.023	6.391	6.897	7.218	7.405	8.451	8.931	9.825	10.810
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	11.431	11.796	11.440	13.075	13.639	10.903	7.939	5.533	6.339	4.431	3.638	3.793
Wednesd	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	3.055	4.379	5.434	6.310	6.569	7.247	7.497	7.931	8.190	9.451	10.517	11.121
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	11.422	11.328	11.106	15.133	14.477	13.113	10.341	7.947	7.402	6.575	5.374	4.402
Thursday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	2.986	4.471	5.310	5.523	5.914	6.572	6.598	8.006	8.675	9.037	10.276	11.368
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	12.805	14.546	15.126	24.294	27.421	24.955	23.315	21.245	20.776	15.230	9.184	6.759
Friday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	3.083	4.233	5.333	5.546	5.764	6.672	6.776	7.080	8.115	8.170	7.968	8.707
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	10.302	10.922	11.348	18.998	21.386	22.036	21.796	18.470	18.391	13.885	9.333	6.069
Saturday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	3.106	3.644	4.302	4.840	5.234	5.787	5.668	6.440	7.278	7.366	8.057	8.810
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	9.670	10.082	9.284	11.888	12.579	10.360	7.489	5.721	5.960	4.347	3.290	3.801

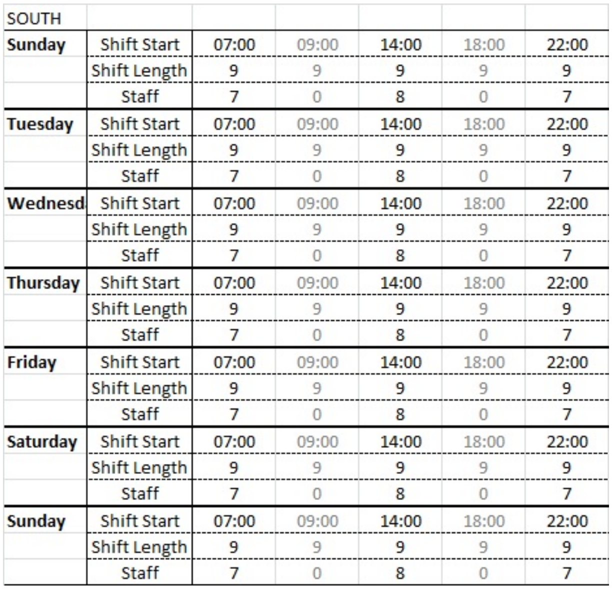


NORTH						
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	9	0	9	0	9
Tuesday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	9	0	9	0	9
Wednesd	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	9	0	9	0	9
Thursday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	9	0	9	0	9
Friday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	9	0	9	0	9
Saturday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	9	0	9	0	9
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	9	0	9	0	9

Figure A-17: Trial problem based upon the North region of Leicestershire.



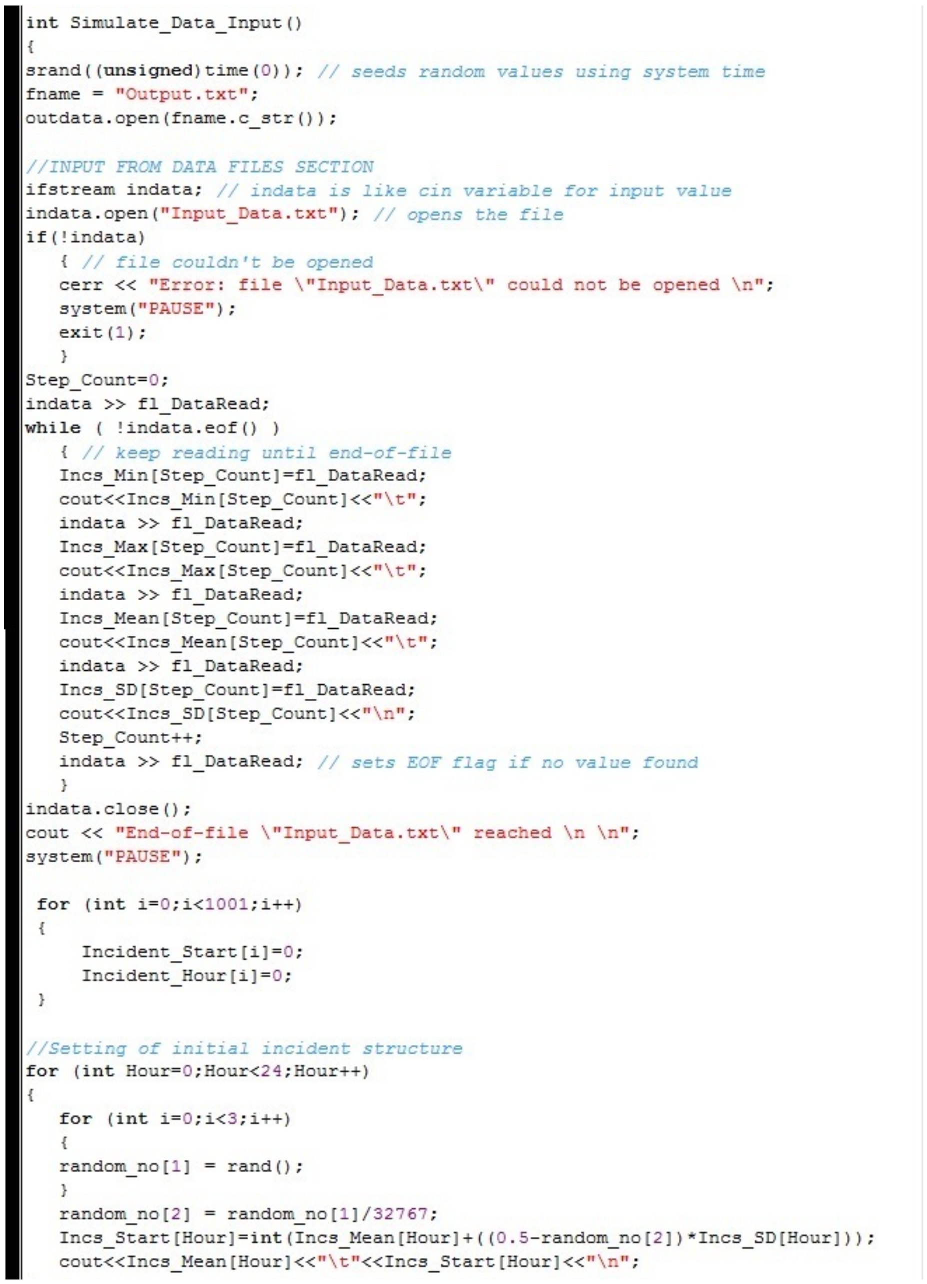
SOUTH													
Sunday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	3.158	4.384	4.836	4.836	5.195	5.690	5.468	6.402	7.055	8.190	8.874	8.934
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	9.670	9.776	8.693	10.280	9.320	8.096	6.229	4.662	4.810	3.684	3.362	3.833
Monday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	2.960	4.402	5.034	5.069	4.905	5.158	5.767	6.675	6.997	7.132	8.914	9.261
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	9.037	8.641	8.730	10.844	10.553	8.087	6.092	4.148	4.052	3.575	3.080	3.351
Tuesday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	3.043	4.124	4.500	4.500	5.509	6.207	6.216	6.601	7.316	7.911	8.767	9.592
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	9.494	10.078	10.259	11.775	10.503	9.338	6.941	4.904	4.489	3.351	3.063	3.603
Wednesd	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	3.109	4.213	5.144	5.230	5.356	5.727	6.299	6.388	7.095	7.405	8.471	8.638
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	9.529	9.046	9.129	11.356	12.101	10.193	8.384	5.828	6.322	5.615	4.316	3.770
Thursday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	3.391	4.687	5.101	5.198	5.684	6.261	6.457	7.072	7.641	8.943	9.138	10.713
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	12.560	12.888	13.198	22.104	23.548	21.587	20.349	15.729	13.029	9.609	7.575	5.776
Friday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	3.086	3.971	4.945	5.230	6.000	6.431	6.445	6.379	6.836	7.445	7.503	7.672
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	8.503	8.905	9.948	16.560	19.411	19.452	18.102	15.748	13.270	10.529	6.747	4.793
Saturday	Time	07:00	08:00	09:00	10:00	11:00	12:00	13:00	14:00	15:00	16:00	17:00	18:00
	Demand	2.618	2.923	3.664	4.268	5.080	5.259	5.304	5.338	5.955	6.716	7.122	7.202
	Time	19:00	20:00	21:00	22:00	23:00	00:00	01:00	02:00	03:00	04:00	05:00	06:00
	Demand	7.702	7.690	7.682	9.782	9.746	8.060	6.223	4.575	4.568	3.784	2.670	3.750



SOUTH						
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	7	0	8	0	7
Tuesday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	7	0	8	0	7
Wednesd	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	7	0	8	0	7
Thursday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	7	0	8	0	7
Friday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	7	0	8	0	7
Saturday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	7	0	8	0	7
Sunday	Shift Start	07:00	09:00	14:00	18:00	22:00
	Shift Length	9	9	9	9	9
	Staff	7	0	8	0	7

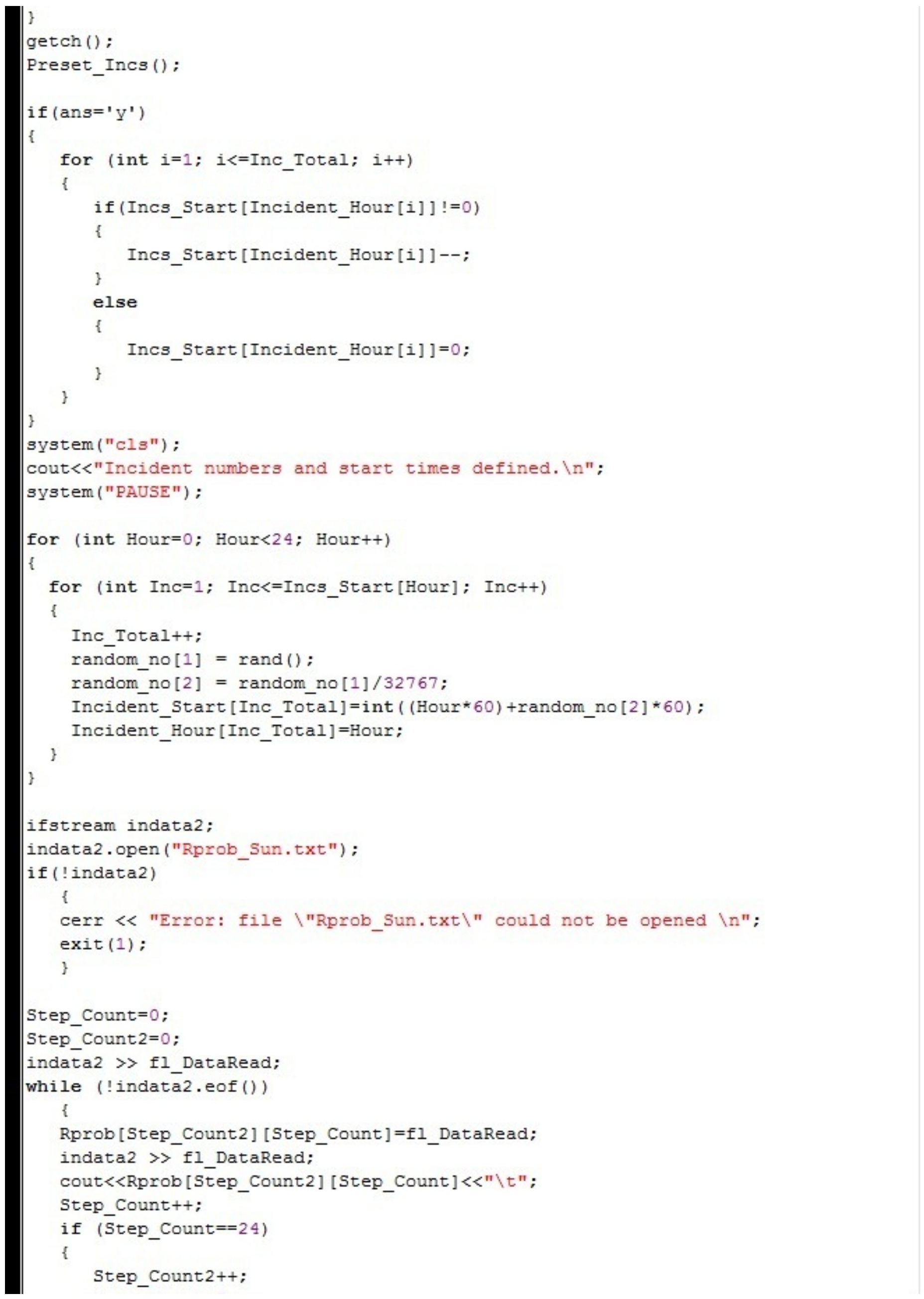
Figure A-18: Trial problem based upon the South region of Leicestershire.

Appendix B. Probabilistic Methodology C++ Code.



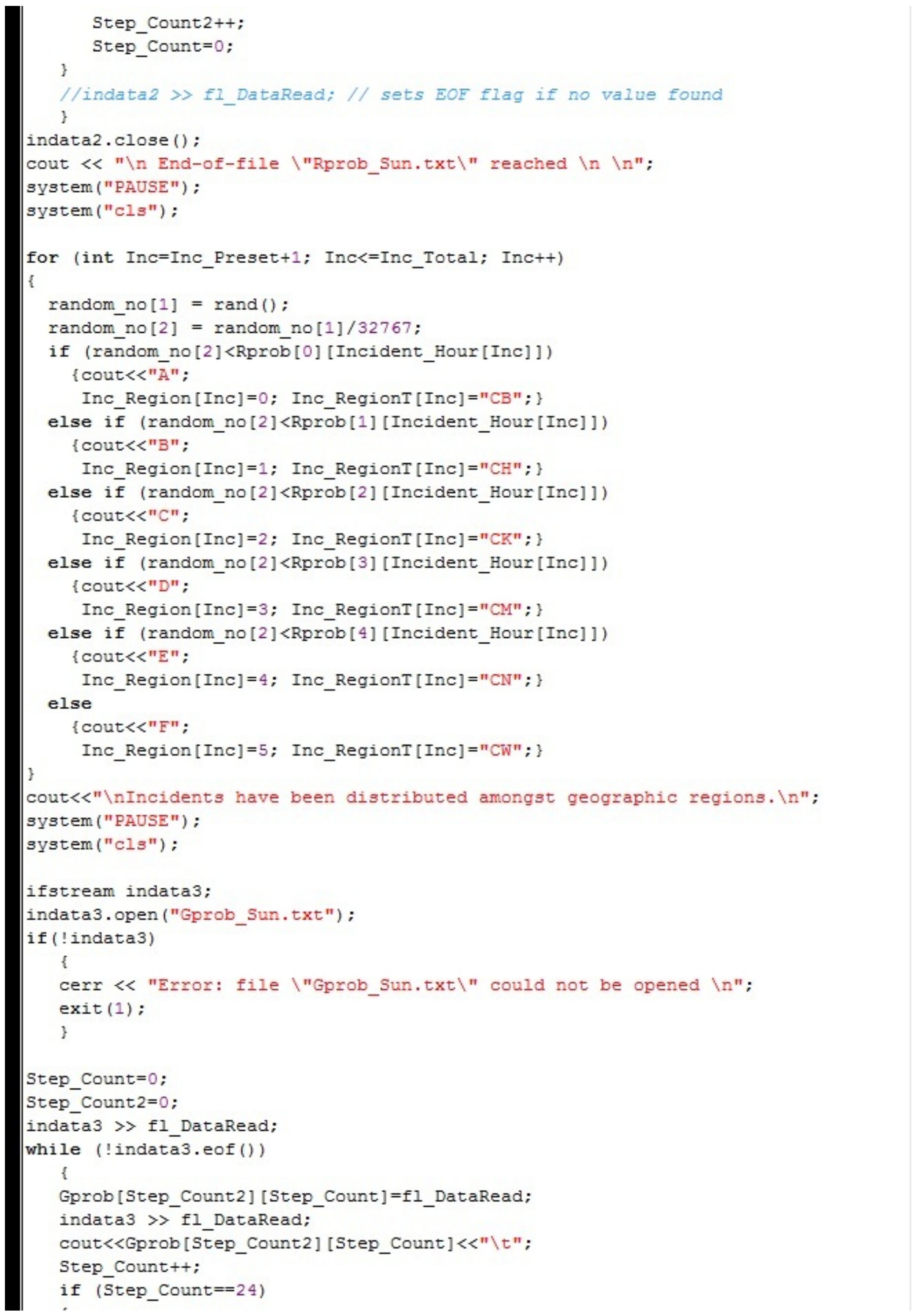
```
int Simulate_Data_Input()
{
srand((unsigned)time(0)); // seeds random values using system time
fname = "Output.txt";
outdata.open(fname.c_str());

//INPUT FROM DATA FILES SECTION
ifstream indata; // indata is like cin variable for input value
indata.open("Input_Data.txt"); // opens the file
if(!indata)
   { // file couldn't be opened
   cerr << "Error: file \"Input_Data.txt\" could not be opened \n";
   system("PAUSE");
   exit(1);
   }
Step_Count=0;
indata >> fl_DataRead;
while ( !indata.eof() )
   { // keep reading until end-of-file
   Incs_Min[Step_Count]=fl_DataRead;
   cout<<Incs_Min[Step_Count]<<"\t";
   indata >> fl_DataRead;
   Incs_Max[Step_Count]=fl_DataRead;
   cout<<Incs_Max[Step_Count]<<"\t";
   indata >> fl_DataRead;
   Incs_Mean[Step_Count]=fl_DataRead;
   cout<<Incs_Mean[Step_Count]<<"\t";
   indata >> fl_DataRead;
   Incs_SD[Step_Count]=fl_DataRead;
   cout<<Incs_SD[Step_Count]<<"\n";
   Step_Count++;
   indata >> fl_DataRead; // sets EOF flag if no value found
   }
indata.close();
cout << "End-of-file \"Input_Data.txt\" reached \n \n";
system("PAUSE");

 for (int i=0;i<1001;i++)
 {
     Incident_Start[i]=0;
     Incident_Hour[i]=0;
 }

//Setting of initial incident structure
for (int Hour=0;Hour<24;Hour++)
{
   for (int i=0;i<3;i++)
   {
   random_no[1] = rand();
   }
   random_no[2] = random_no[1]/32767;
   Incs_Start[Hour]=int(Incs_Mean[Hour]+((0.5-random_no[2])*Incs_SD[Hour]));
   cout<<Incs_Mean[Hour]<<"\t"<<Incs_Start[Hour]<<"\n";
```

Figure B-19: Page 1 of probabilistic methodology C++ code.



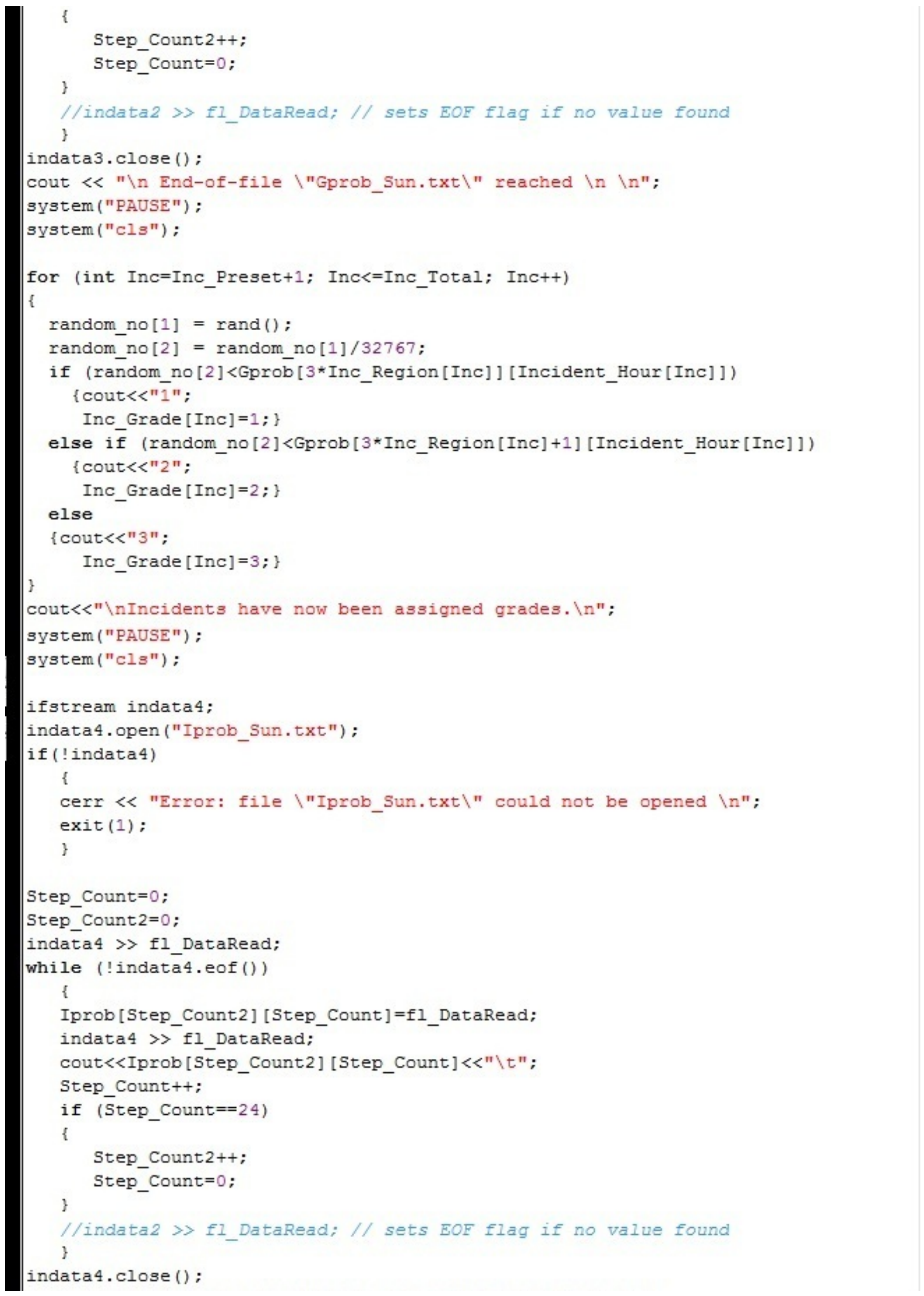
```
}
getch();
Preset_Incs();

if(ans='y')
{
   for (int i=1; i<=Inc_Total; i++)
   {
      if(Incs_Start[Incident_Hour[i]]!=0)
      {
         Incs_Start[Incident_Hour[i]]--;
      }
      else
      {
         Incs_Start[Incident_Hour[i]]=0;
      }
   }
}
system("cls");
cout<<"Incident numbers and start times defined.\n";
system("PAUSE");

for (int Hour=0; Hour<24; Hour++)
{
  for (int Inc=1; Inc<=Incs_Start[Hour]; Inc++)
  {
    Inc_Total++;
    random_no[1] = rand();
    random_no[2] = random_no[1]/32767;
    Incident_Start[Inc_Total]=int((Hour*60)+random_no[2]*60);
    Incident_Hour[Inc_Total]=Hour;
  }
}

ifstream indata2;
indata2.open("Rprob_Sun.txt");
if(!indata2)
   {
   cerr << "Error: file \"Rprob_Sun.txt\" could not be opened \n";
   exit(1);
   }

Step_Count=0;
Step_Count2=0;
indata2 >> fl_DataRead;
while (!indata2.eof())
   {
   Rprob[Step_Count2][Step_Count]=fl_DataRead;
   indata2 >> fl_DataRead;
   cout<<Rprob[Step_Count2][Step_Count]<<"\t";
   Step_Count++;
   if (Step_Count==24)
   {
      Step_Count2++;
```

Figure B-20: Page 2 of probabilistic methodology C++ code.



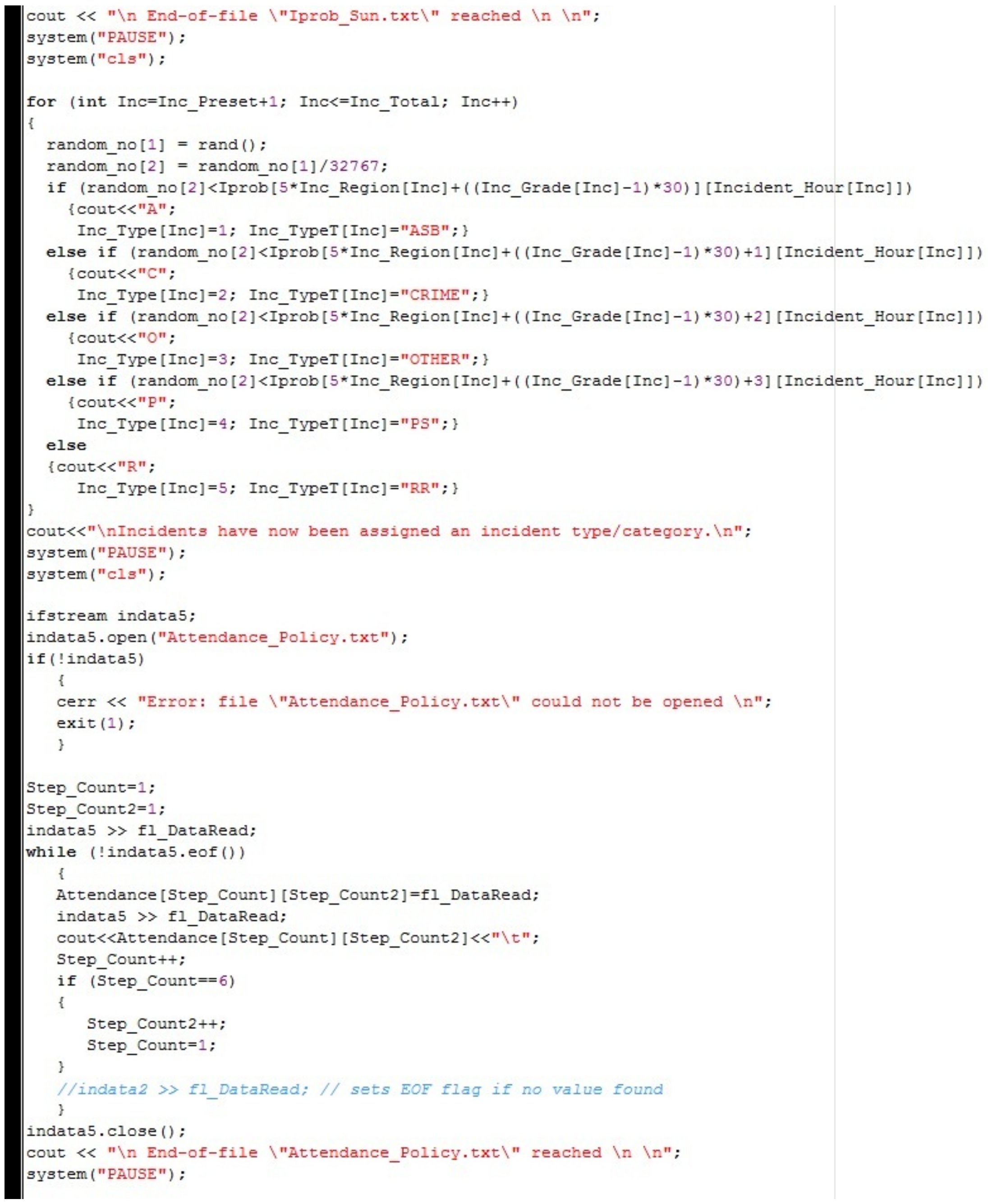
```
      Step_Count2++;
      Step_Count=0;
    }
    //indata2 >> fl_DataRead; // sets EOF flag if no value found
    }
indata2.close();
cout << "\n End-of-file \"Rprob_Sun.txt\" reached \n \n";
system("PAUSE");
system("cls");

for (int Inc=Inc_Preset+1; Inc<=Inc_Total; Inc++)
{
  random_no[1] = rand();
  random_no[2] = random_no[1]/32767;
  if (random_no[2]<Rprob[0][Incident_Hour[Inc]])
    {cout<<"A";
     Inc_Region[Inc]=0; Inc_RegionT[Inc]="CB";}
  else if (random_no[2]<Rprob[1][Incident_Hour[Inc]])
    {cout<<"B";
     Inc_Region[Inc]=1; Inc_RegionT[Inc]="CH";}
  else if (random_no[2]<Rprob[2][Incident_Hour[Inc]])
    {cout<<"C";
     Inc_Region[Inc]=2; Inc_RegionT[Inc]="CK";}
  else if (random_no[2]<Rprob[3][Incident_Hour[Inc]])
    {cout<<"D";
     Inc_Region[Inc]=3; Inc_RegionT[Inc]="CM";}
  else if (random_no[2]<Rprob[4][Incident_Hour[Inc]])
    {cout<<"E";
     Inc_Region[Inc]=4; Inc_RegionT[Inc]="CN";}
  else
    {cout<<"F";
     Inc_Region[Inc]=5; Inc_RegionT[Inc]="CW";}
}
cout<<"\nIncidents have been distributed amongst geographic regions.\n";
system("PAUSE");
system("cls");

ifstream indata3;
indata3.open("Gprob_Sun.txt");
if(!indata3)
    {
    cerr << "Error: file \"Gprob_Sun.txt\" could not be opened \n";
    exit(1);
    }

Step_Count=0;
Step_Count2=0;
indata3 >> fl_DataRead;
while (!indata3.eof())
    {
    Gprob[Step_Count2][Step_Count]=fl_DataRead;
    indata3 >> fl_DataRead;
    cout<<Gprob[Step_Count2][Step_Count]<<"\t";
    Step_Count++;
    if (Step_Count==24)
```

Figure B-21: Page 3 of probabilistic methodology C++ code.



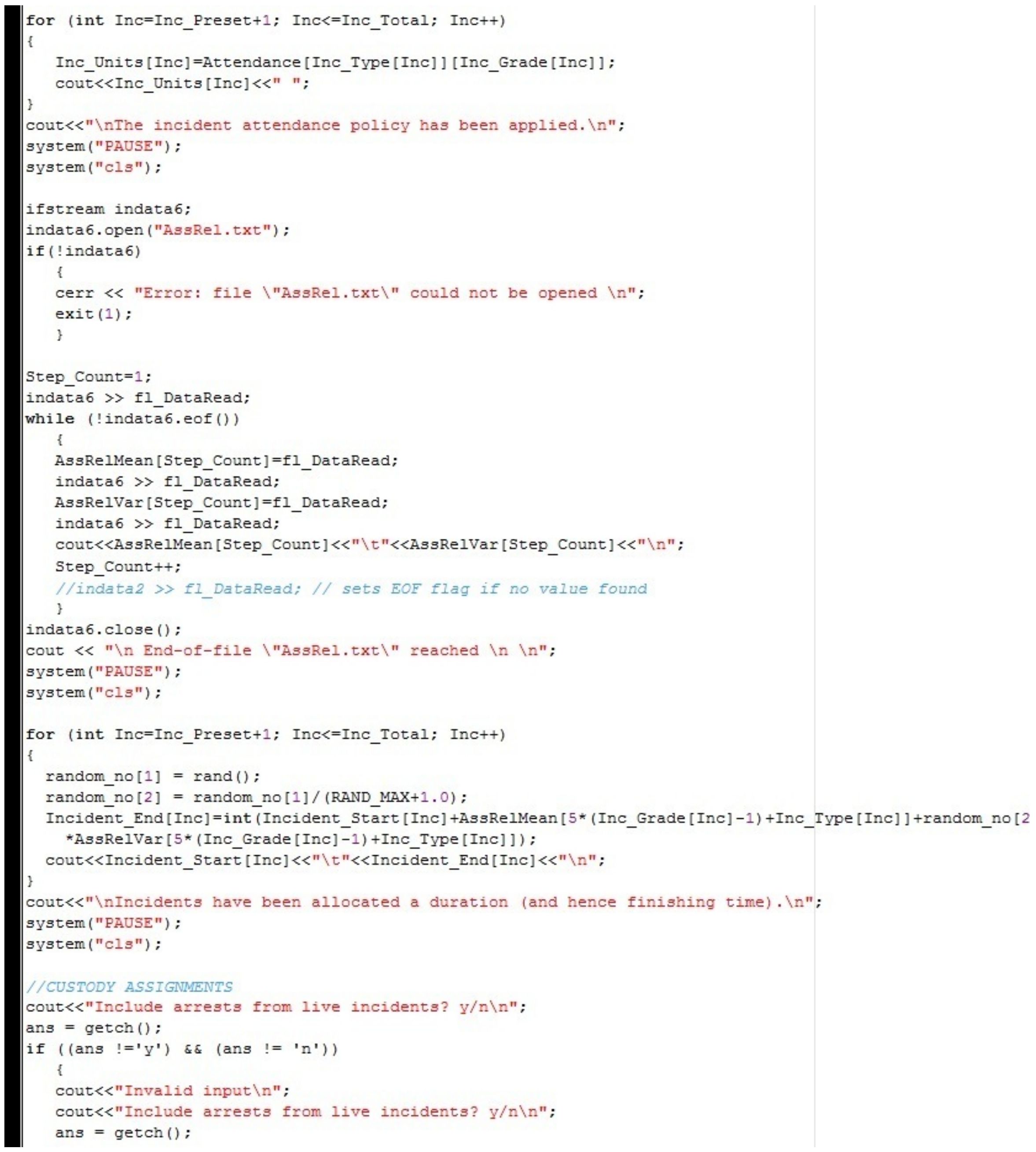
```
    {
       Step_Count2++;
       Step_Count=0;
    }
    //indata2 >> fl_DataRead; // sets EOF flag if no value found
    }
indata3.close();
cout << "\n End-of-file \"Gprob_Sun.txt\" reached \n \n";
system("PAUSE");
system("cls");

for (int Inc=Inc_Preset+1; Inc<=Inc_Total; Inc++)
{
  random_no[1] = rand();
  random_no[2] = random_no[1]/32767;
  if (random_no[2]<Gprob[3*Inc_Region[Inc]][Incident_Hour[Inc]])
    {cout<<"1";
     Inc_Grade[Inc]=1;}
  else if (random_no[2]<Gprob[3*Inc_Region[Inc]+1][Incident_Hour[Inc]])
    {cout<<"2";
     Inc_Grade[Inc]=2;}
  else
  {cout<<"3";
     Inc_Grade[Inc]=3;}
}
cout<<"\nIncidents have now been assigned grades.\n";
system("PAUSE");
system("cls");

ifstream indata4;
indata4.open("Iprob_Sun.txt");
if(!indata4)
    {
    cerr << "Error: file \"Iprob_Sun.txt\" could not be opened \n";
    exit(1);
    }

Step_Count=0;
Step_Count2=0;
indata4 >> fl_DataRead;
while (!indata4.eof())
    {
    Iprob[Step_Count2][Step_Count]=fl_DataRead;
    indata4 >> fl_DataRead;
    cout<<Iprob[Step_Count2][Step_Count]<<"\t";
    Step_Count++;
    if (Step_Count==24)
    {
       Step_Count2++;
       Step_Count=0;
    }
    //indata2 >> fl_DataRead; // sets EOF flag if no value found
    }
indata4.close();
```

Figure B-22: Page 4 of probabilistic methodology C++ code.



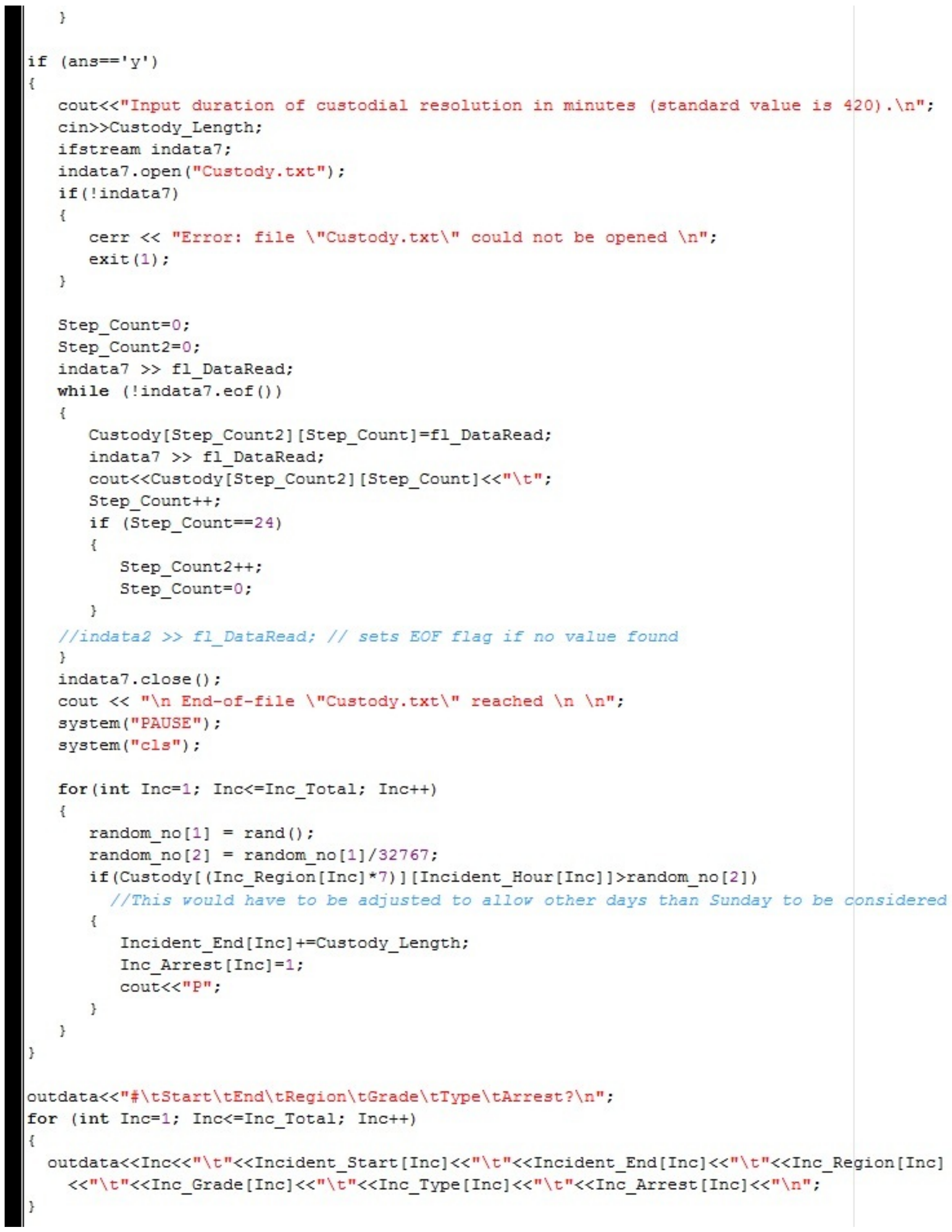
```
cout << "\n End-of-file \"Iprob_Sun.txt\" reached \n \n";
system("PAUSE");
system("cls");

for (int Inc=Inc_Preset+1; Inc<=Inc_Total; Inc++)
{
  random_no[1] = rand();
  random_no[2] = random_no[1]/32767;
  if (random_no[2]<Iprob[5*Inc_Region[Inc]+((Inc_Grade[Inc]-1)*30)][Incident_Hour[Inc]])
    {cout<<"A";
     Inc_Type[Inc]=1; Inc_TypeT[Inc]="ASB";}
  else if (random_no[2]<Iprob[5*Inc_Region[Inc]+((Inc_Grade[Inc]-1)*30)+1][Incident_Hour[Inc]])
    {cout<<"C";
     Inc_Type[Inc]=2; Inc_TypeT[Inc]="CRIME";}
  else if (random_no[2]<Iprob[5*Inc_Region[Inc]+((Inc_Grade[Inc]-1)*30)+2][Incident_Hour[Inc]])
    {cout<<"O";
     Inc_Type[Inc]=3; Inc_TypeT[Inc]="OTHER";}
  else if (random_no[2]<Iprob[5*Inc_Region[Inc]+((Inc_Grade[Inc]-1)*30)+3][Incident_Hour[Inc]])
    {cout<<"P";
     Inc_Type[Inc]=4; Inc_TypeT[Inc]="PS";}
  else
  {cout<<"R";
     Inc_Type[Inc]=5; Inc_TypeT[Inc]="RR";}
}
cout<<"\nIncidents have now been assigned an incident type/category.\n";
system("PAUSE");
system("cls");

ifstream indata5;
indata5.open("Attendance_Policy.txt");
if(!indata5)
   {
   cerr << "Error: file \"Attendance_Policy.txt\" could not be opened \n";
   exit(1);
   }

Step_Count=1;
Step_Count2=1;
indata5 >> fl_DataRead;
while (!indata5.eof())
   {
   Attendance[Step_Count][Step_Count2]=fl_DataRead;
   indata5 >> fl_DataRead;
   cout<<Attendance[Step_Count][Step_Count2]<<"\t";
   Step_Count++;
   if (Step_Count==6)
   {
      Step_Count2++;
      Step_Count=1;
   }
   //indata2 >> fl_DataRead; // sets EOF flag if no value found
   }
indata5.close();
cout << "\n End-of-file \"Attendance_Policy.txt\" reached \n \n";
system("PAUSE");
```

Figure B-23: Page 5 of probabilistic methodology C++ code.



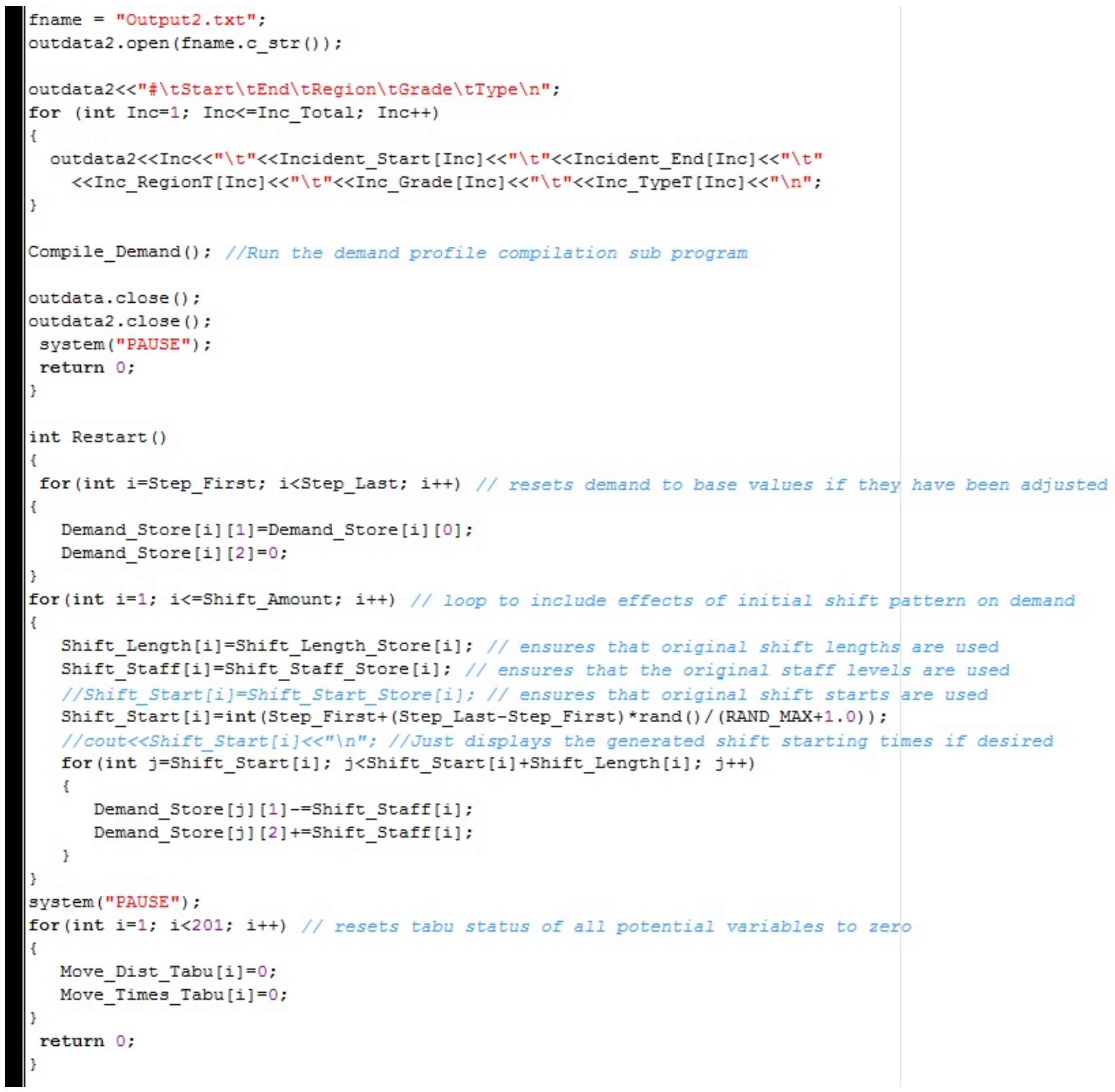
```
for (int Inc=Inc_Preset+1; Inc<=Inc_Total; Inc++)
{
   Inc_Units[Inc]=Attendance[Inc_Type[Inc]][Inc_Grade[Inc]];
   cout<<Inc_Units[Inc]<<" ";
}
cout<<"\nThe incident attendance policy has been applied.\n";
system("PAUSE");
system("cls");

ifstream indata6;
indata6.open("AssRel.txt");
if(!indata6)
   {
   cerr << "Error: file \"AssRel.txt\" could not be opened \n";
   exit(1);
   }

Step_Count=1;
indata6 >> fl_DataRead;
while (!indata6.eof())
   {
   AssRelMean[Step_Count]=fl_DataRead;
   indata6 >> fl_DataRead;
   AssRelVar[Step_Count]=fl_DataRead;
   indata6 >> fl_DataRead;
   cout<<AssRelMean[Step_Count]<<"\t"<<AssRelVar[Step_Count]<<"\n";
   Step_Count++;
   //indata2 >> fl_DataRead; // sets EOF flag if no value found
   }
indata6.close();
cout << "\n End-of-file \"AssRel.txt\" reached \n \n";
system("PAUSE");
system("cls");

for (int Inc=Inc_Preset+1; Inc<=Inc_Total; Inc++)
{
  random_no[1] = rand();
  random_no[2] = random_no[1]/(RAND_MAX+1.0);
  Incident_End[Inc]=int(Incident_Start[Inc]+AssRelMean[5*(Inc_Grade[Inc]-1)+Inc_Type[Inc]]+random_no[2
    *AssRelVar[5*(Inc_Grade[Inc]-1)+Inc_Type[Inc]]);
  cout<<Incident_Start[Inc]<<"\t"<<Incident_End[Inc]<<"\n";
}
cout<<"\nIncidents have been allocated a duration (and hence finishing time).\n";
system("PAUSE");
system("cls");

//CUSTODY ASSIGNMENTS
cout<<"Include arrests from live incidents? y/n\n";
ans = getch();
if ((ans !='y') && (ans != 'n'))
   {
   cout<<"Invalid input\n";
   cout<<"Include arrests from live incidents? y/n\n";
   ans = getch();
```

Figure B-24: Page 6 of probabilistic methodology C++ code.



```
   }

if (ans=='y')
{
   cout<<"Input duration of custodial resolution in minutes (standard value is 420).\n";
   cin>>Custody_Length;
   ifstream indata7;
   indata7.open("Custody.txt");
   if(!indata7)
   {
      cerr << "Error: file \"Custody.txt\" could not be opened \n";
      exit(1);
   }

   Step_Count=0;
   Step_Count2=0;
   indata7 >> fl_DataRead;
   while (!indata7.eof())
   {
      Custody[Step_Count2][Step_Count]=fl_DataRead;
      indata7 >> fl_DataRead;
      cout<<Custody[Step_Count2][Step_Count]<<"\t";
      Step_Count++;
      if (Step_Count==24)
      {
         Step_Count2++;
         Step_Count=0;
      }
   //indata2 >> fl_DataRead; // sets EOF flag if no value found
   }
   indata7.close();
   cout << "\n End-of-file \"Custody.txt\" reached \n \n";
   system("PAUSE");
   system("cls");

   for(int Inc=1; Inc<=Inc_Total; Inc++)
   {
      random_no[1] = rand();
      random_no[2] = random_no[1]/32767;
      if(Custody[(Inc_Region[Inc]*7)][Incident_Hour[Inc]]>random_no[2])
        //This would have to be adjusted to allow other days than Sunday to be considered
      {
         Incident_End[Inc]+=Custody_Length;
         Inc_Arrest[Inc]=1;
         cout<<"P";
      }
   }
}

outdata<<"#\tStart\tEnd\tRegion\tGrade\tType\tArrest?\n";
for (int Inc=1; Inc<=Inc_Total; Inc++)
{
  outdata<<Inc<<"\t"<<Incident_Start[Inc]<<"\t"<<Incident_End[Inc]<<"\t"<<Inc_Region[Inc]
    <<"\t"<<Inc_Grade[Inc]<<"\t"<<Inc_Type[Inc]<<"\t"<<Inc_Arrest[Inc]<<"\n";
}
```

Figure B-25: Page 7 of probabilistic methodology C++ code.



```
fname = "Output2.txt";
outdata2.open(fname.c_str());

outdata2<<"#\tStart\tEnd\tRegion\tGrade\tType\n";
for (int Inc=1; Inc<=Inc_Total; Inc++)
{
  outdata2<<Inc<<"\t"<<Incident_Start[Inc]<<"\t"<<Incident_End[Inc]<<"\t"
    <<Inc_RegionT[Inc]<<"\t"<<Inc_Grade[Inc]<<"\t"<<Inc_TypeT[Inc]<<"\n";
}

Compile_Demand(); //Run the demand profile compilation sub program

outdata.close();
outdata2.close();
 system("PAUSE");
 return 0;
}

int Restart()
{
 for(int i=Step_First; i<Step_Last; i++) // resets demand to base values if they have been adjusted
{
   Demand_Store[i][1]=Demand_Store[i][0];
   Demand_Store[i][2]=0;
}
for(int i=1; i<=Shift_Amount; i++) // loop to include effects of initial shift pattern on demand
{
   Shift_Length[i]=Shift_Length_Store[i]; // ensures that original shift lengths are used
   Shift_Staff[i]=Shift_Staff_Store[i]; // ensures that the original staff levels are used
   //Shift_Start[i]=Shift_Start_Store[i]; // ensures that original shift starts are used
   Shift_Start[i]=int(Step_First+(Step_Last-Step_First)*rand()/(RAND_MAX+1.0));
   //cout<<Shift_Start[i]<<"\n"; //Just displays the generated shift starting times if desired
   for(int j=Shift_Start[i]; j<Shift_Start[i]+Shift_Length[i]; j++)
   {
      Demand_Store[j][1]-=Shift_Staff[i];
      Demand_Store[j][2]+=Shift_Staff[i];
   }
}
system("PAUSE");
for(int i=1; i<201; i++)  // resets tabu status of all potential variables to zero
{
   Move_Dist_Tabu[i]=0;
   Move_Times_Tabu[i]=0;
}
 return 0;
}
```

Figure B-26: Page 8 of probabilistic methodology C++ code.

This Page Left Blank

References

[1] Home Office *Quality of Service Commitment. Unpublished*

[2] Guide, D. Teunter, R. & Wassenhove , L. *Matching Demand and Supply to Maximize Profits from Remanufacturing. Manufacturing & Service Operations Management Vol. 5 No. 4 (2003) pp. 303-316*

[3] Nozick, L. & Turnquist, M. *Inventory, transportation, service quality and the location of distribution centers. European Journal of Operational Research Vol. 129 (2001) pp. 362-371*

[4] Giri, B. & Chaudhuria, K. *Deterministic models of perishable inventory with stock-dependent demand rate and nonlinear holding cost. European Journal of Operational Research Vol. 105 (1998) pp. 467-474*

[5] Baker, R. & Urban, T. *A Deterministic Inventory System with an Inventory-Level-Dependent Demand Rate. The Journal of the Operational Research Society Vol. 39 No. 9 (1988) pp. 823-831*

[6] Fishman, G. *Principles of Discrete Event Simulation. ISBN 0471043958*

[7] Bonabeau, E. *Agent-based modeling: Methods and techniques for simulating human systems. PNAS Vol. 99 (2002) pp. 7280-7287*

[8] Glover, F. & Laguna, M. *Tabu Search. ISBN 0-7923-8187-4.*

[9] Burke, E.K. Causmaecker, P.De and Berghe, G.V. *A Hybrid Tabu Search Algorithm for the Nurse Rostering Problem. Lecture Notes in Computer Science Vol. 1585 (1999) pp. 187-194*

[10] Gendreau, M. Hertz, A. & Laporte, G. *A Tabu Search Heuristic for the Vehicle Routing Problem. Management Science Vol. 40 No. 10 (Oct. 1994) pp. 1276-1290*

[11] Dowsland, K. *Nurse scheduling with tabu search and strategic oscillation. European Journal of Operational Research Vol. 106 (1998) pp. 393-407*

[12] Murata, T. Ishibuchi, H. & Tanaka, H. *Genetic algorithms for flowshop scheduling problems. Computers & Industrial Engineering Vol. 30 No. 4 (Sep. 1996) pp. 1061-1071*

[13] Haupt, R. & Haupt, S.E. *Practical Genetic Algorithms. John Wiley & Sons. 1998*

[14] Hu, X.B. & Paolo, E. *An Efficient Genetic Algorithm with Uniform Crossover for the Multi-Objective Airport Gate Assignment Problem. Multi-Objective Memetic Algorithms. in press*

[15] Glover, F. Kelly, J. & Lagune, M. *Genetic algorithms and tabu search: Hybrids for optimization. Computers & operations Research Vol. 22 No. 1 (Jan. 1995) pp. 111-134*

[16] Fleurent, C. & Ferland, J. *Genetic and hybrid algorithms for graph coloring. Annals of Operations Research Vol. 63 No. 3 pp. 437-461*

[17] *Genetic Tabu Hybrid Algorithm and its Applications in Power Network Planning. Automation of Electric Power Systems 2004-20*

[18] Shumway, R. *Supply, Demand, and Technology in a Multiproduct Industry: Texas Field Crops. American Journal of Agricultural Economics Vol. 65 No. 4 (1983) pp. 748-760*

[19] Dick, A. *Demand estimation and consumer welfare in the banking industry. Journal of Banking & Finance Vol. 32 (2008) pp. 1661-1676*

[20] Duan, N. Manning, W. Morris, C. & Newhouse, P. *A Comparison of Alternative Models for the Demand for Medical Care. Journal of Business & Economic Statistics Vol. 1 No. 2 (1983) pp. 115-126*

[21] Burke, E.K. Kendall, G. & Soubeiga, E. *A Tabu-Search Hyperheuristic for Timetabling and Rostering. Journal of Heuristics Vol. 9 (2003) pp. 451-470.*

[22] Burke, E. Cowling, P. Causmaecker, P. & Berghe, G. *A Memetic Approach to the Nurse Rostering Problem. Applied Intelligence Vol. 15 No. 3 (2001) pp. 199-214*

[23] Cheanga, B. Lib ,H. Lim, A. & Rodrigues, B. *Nurse rostering problems: a bibliographic survey. European Journal of Operational Research Vol. 151 (2003) pp. 447-460*

[24] Burke, E.K. Causmaecker, P.D. & Berghe, G.V. *A Hybrid Tabu Search Algorithm for the Nurse Rostering Problem. Lecture Notes in Computer Science Vol. 1585 (1999) pp. 187-194*

[25] Jiang, Y. Kautz, H. & Selman, B. *Solving Problems with Hard and Soft Constraints Using a Stochastic Algorithm for MAX-SAT. 1st International Joint Workshop on Artificial Intelligence and Operations Research, Timberline, Oregon (1995)*

[26] Naudin, E. Chan, P. Hiroux, M. Zemmouri, T. & Weil, G. *Analysis of three mathematical models of the Staff Rostering Problem. Journal of Scheduling (2009)*

[27] Greasley, A. & Barlow, S. *Using simulation modelling for BPR: resource allocation in a Police custody process. International Journal of Operations & Production Management Vol. 18 (1998) pp. 978-988*

[28] Reeves, C. *A genetic algorithm for flowshop sequencing. Computers & Operations Research Vol. 22 No. 1 (Jan. 1995) pp. 5-13*

[29] Silver, E.A. *Operations Research in Inventory Management: A Review and Critique. Operations Research Vol. 29 No. 4 (Jul. 1981) pp. 628-645*

[30] Beamon, B. *Supply chain design and analysis:: Models and methods. International Journal of Production Economics Vol.550 (1998) pp. 281-294*

[31] Cachon, G. & Fisher, M. *Supply Chain Inventory Management and the Value of Shared Information. Management Science Vol. 46 No. 8 (2000) pp. 1032-1048*

[32] Vaessens, R. Aarts, E. & Lenstra, J. *Job Shop Scheduling by Local Search. INFORMS Journal on Computing Vol. 8 (1994) pp. 302-317*

[33] Ahujaa, R. Ergunb, O. Orlinc, J. & Punnend, A. *A survey of very large-scale neighborhood search techniques. Discrete Applied Mathematics Vol. 123 (2002) pp. 75-102*

[34] Seckiner, S. & Kurt, M. *A simulated annealing approach to the solution of job rotation scheduling problems. Applied maths and computation Vol. 188 (2007) pp. 31-45*

[35] Carnahan, B. Redfern, M. & Norman, B. *Designing safe job rotation schedules using optimization and heuristic search. Ergonomics Vol. 43 (2000) pp. 543-560*

[36] Dorigo, M. *Optimization, Learning and Natural Algorithm. Politecnico di Milano, Italie, (1992)*

[37] Bedi, P. Mediratta, N. Dhand, S. Sharma, R. & Singhal, A. *Avoiding Traffic Jam Using Ant Colony Optimization - A Novel Approach. International Conference on Computational Intelligence and Multimedia Applications (2007)*

[38] Bell, J. & McMullen, P. *Ant colony optimization techniques for the vehicle routing problem. Advanced Engineering Informatics Vol. 18 (Jan. 2004) pp. 41-48*

[39] Gravel, M. Price, W. & Gagne, C. *Scheduling continuous casting of aluminium using a multiple objective ant colony optimization metaheuristic. European Journal of Operational Research Vol. 143 (2002) pp. 218-229*

[40] Ying, K. & Liao, C. *An ant-colony system for permutation flow-shop sequencing. Computers and operations research Vol. 31 (2004) pp. 791-801*

[41] Income Data Services *Working Time, Employment Law Handbook. ISBN 0308-9312*

[42] Bowman, A. & Robinson, D. *Introduction to Statistics. ISBN 978-0852744086*

[43] Glover, F. *Tabu search for nonlinear and parametric optimization. Discrete Applied Mathematics Vol. 49 (Mar. 1994) pp. 231-255.*

[44] Musliu, N. Schaerf, A. & Slany, W. *Local Search for Shift Design. European Journal of Operational Research Vol. 153 (2004) pp. 51-64*

[45] Gaspero, D. Gartner, J. Kortsarz, G. Musliu, N. Schaerf, A. & Slany, W. *The Minimum Shift Design Problem. Annals of Operational Research Vol. 155 (2007) pp. 79-105*

[46] Jones, G. Willett, P. Glen, R. Leach, A. & Taylor, R. *Development and validation of a genetic algorithm for flexible docking. Journal of Molecular Biology Vol. 267 (1997) pp. 727-748*

[47] Srinivas, M. & Patnaik, L. *Adaptive probabilities of crossover and mutation in genetic algorithms. Systems, Man and Cybernetics Vol. 20 (1994) pp. 656-667*

[48] Goldberg, D. & Deb, K. *A comparative analysis of selection schemes used in genetic algorithms. Foundations of Genetic Algorithms Vol. 1 pp. 69-92*

[49] Moz, M. & Pato, M. *A genetic algorithm approach to a nurse re-rostering problem. Computers & Operations Research Vol. 34 (2007) pp. 667-691*

[50] Ernst, A. Jiang, H. Krishnamoorthy, M. & Sier, D. *Staff scheduling and rostering: A review of applications, methods and models. European Journal of Operational Research Vol. 153 (2004) pp. 3-27*

[51] Reeves, C. *Modern Heuristic Techniques for Combinatorial Problems. ISBN 0-07-709239-2*

[52] Glover, F. *Tabu Search Part I. INFORMS Journal on Computing Vol. 1 No. 3 (1989) pp. 190-206.*

[53] Glover, F. *Tabu Search Part II. INFORMS Journal on Computing Vol. 2 No. 1 (1989) pp. 4-32.*

[54] Holland, J.H. *Adaptation in Natural and Artificial Systems. ISBN 0-26-258111-6*

[55] *http://www.esteco.com/products.jsp*

[56] Thangiah, S. Osman, I. & Sun, T. *Hybrid genetic algorithm, simulated annealing and tabu search methods for vehicle routing problems with time windows.*

[57] Goncalves, J. Mendes, J. & Resende, M. *A Hybrid Genetic Algorithm for the Job Shop Scheduling Problem.*

Appendix C. Published Research Paper.

A TABU SEARCH ALGORITHM APPLIED TO THE STAFFING ROSTER PROBLEM OF LEICESTERSHIRE POLICE FORCE

Mr O.S.S.T Edleston & Dr L.M Bartlett

Aeronautical and Automotive Engineering Dept.

Loughborough University, Leicestershire

ABSTRACT

This paper presents an application of the tabu search algorithm to a staff rostering problem relevant to Leicestershire Police. The aim is to address the issue of structuring staff rosters to enable effective use of staff to meet the demand on the Police to reduce and deal with crime related incidents. This problem is defined through the compilation of a time varying level of required staff and an associated staff roster. The objective is an optimised work set up, maximising staff resources and the meeting of demand. Optimisation of staff levels to demand is sought through use of a series of tabu search algorithms, making use of two diversification techniques and an intensification technique individually and in compilation. The tabu search is shown to be a well suited optimisation approach to the type of problem defined, with individual conclusions drawn for each of the technique combinations used.

Keywords: tabu search, staff rostering, Police, optimisation

1 INTRODUCTION

A common problem faced by many industries and organisations is how best to use or supply available resources in order to meet a quantifiable demand; this is true not only for the supply of material stock items but also when considering the distribution of staff members over time. A wide variety of staffing roster problems are investigated both within industry and the literature. Within this work, problems are considered where there exist time dependent demands which are used to quantify the required number of staff members on duty over time. This type of problem is commonly referred to as an operator staffing problem (Jennings et al., 1996), (Baker, 1976). Within the work of Baker (Baker, 1976) a survey of mathematical models that have been used in order to solve such problems is provided; however the work focuses on situations in which demand is cyclical, with no strong fluctuations such as those that are experienced within the Police problem. Jennings et al. (Jennings et al., 1996) define a method for calculating a total staffing (or server) amount required in order to increase the probability of staff availability to meet any newly occurring demand above a target percentage. The problem posed by Leicestershire Police differs from this however in that it is desired instead to provide the specific arrangement of staff required in order to best meet demand. The most directly related problem to that considered in this paper is the well defined and thoroughly examined nurse rostering problem (Burke et al. 1999, 2003); this problem being one that also considers demands and the provision of staff within a public service business. The work conducted in this field generally focuses upon solution of pre-constructed staffing problems through use of optimisation and iterative search procedures. For example, the work conducted by Burke et al. (Burke et al. 1999) employs the tabu search algorithm for nurse roster optimisation whilst considering an objective function based upon staff satisfaction levels. In their work it is shown that tabu search is a highly suitable optimising tool for such applications and it is used in the development of a software package titled PLANE, for use by National Health Service hospitals in Belgium. In the work of Naudin et al. (Naudin et al., 2009) consider a general staff rostering problem and construct 3 models leading to its solution, these models each consider a different level of tasking that is assigned to each staff member. The first seeks to assign individual tasks on a moment by moment basis to staff, the second considers instead the assignation of daily rosters as a key variable and finally the third considers weekly rosters. A branch and bound methodology is applied to each of these models and conclusions are drawn about the relative strengths and weaknesses of each.

For research of this nature there are many techniques to potentially use. Hertz and Werra (Hertz et al., 1990) provide a general overview of the tabu search and describe application to a variety of optimisation problems. They conclude that tabu search is a valid and powerful optimisation technique which is suited for use in many problems. Gaspero and Schaerf (Gaspero et al., 2001) discuss the use of tabu search upon a particular problem, the scheduling of University examination timetables. A number of constraints are defined which are used for both the invalidation of particular arrangements of examinations or penalisation of certain patterns. They compare the effectiveness of the constructed tabu search upon public benchmarks and note that it performs well against other optimisation techniques. Indeed the tabu search has been successfully applied to well known optimisation problems, such as in the work of Hertz and Werra (Hertz et al., 1987) where, in an approach to the graph colouring problem, it is shown not only that tabu search is well suited but that it even outperforms the simulated annealing technique. Ernst et al. (Ernst et al., 2004) provide a review of the types of rostering problems that have been previously investigated within the literature with particular reference given to individual fields in which previous research has been applied. An alternative objective to that considered in this research is presented in the work of Gaspero et al. (Gaspero et al., 2004) where minimisation of the total required number of shifts is sought. The intent of this is to reduce the complexity present within a series of defined shifts in order to increase the ability with which it may be managed. In contrast, the research presented here considers a fixed number of shifts and seeks to find a best defined series of shifts with a best fit of available staff across them. In the work conducted by Musliu et al. (Musliu et al. 2004) a tabu search method is implemented that redefines the shift structure iteratively in order to best satisfy a given objective function. They show that such a process is of strong benefit to staff rostering problems and demonstrate its effectiveness on both generated and real world problems. Gaspero et al. (Gaspero et al. 2007) demonstrate an alternative objective to that considered here, instead seeking to generate staffing rosters with as few shift variations as possible. The intent of this is to reduce the complexity present within a series of defined shifts in order to increase the ability with which it may be managed. In contrast, the research presented here considers a fixed number of shifts and seeks to best fit available staff across them.

In this research, first the optimisation problem facing Police personnel in allocating staff in order to meet some predicted demand will be defined. Following this, detail will be given of a tabu search algorithm constructed in order to attempt optimisation of this problem. An introduction to diversification and intensification strategies is provided alongside description of their applications in enhancement of the base tabu search algorithm used. Trial problems for optimisation are defined and the constructed search algorithm is applied to these; performance of the algorithm is monitored through use of an objective function that quantifies the surplus demand across a time period of interest. Conclusions will be drawn about the success of the tabu search when applied to the defined problem; different combinations of diversification and intensification strategies are tested with comparisons made between the performances of each. Finally a few suggestions will be made of potential developments that could further enhance investigation into the field of work considered by this paper.

2 TABU SEARCH

2.1 Overview

The tabu search is a very powerful optimisation tool (Glover et al., 1997), with a wide range of applications and a high rate of success due to the method allowing a high level of customisation in order to suit individual needs. For example it has been used to great effect in the problems of staff rostering (Burke et al., 2003) and vehicle routing (Gendreau et al., 1994) in addition to more general optimisation tasks (Glover, 1994). In each of these the tabu search method is shown to compare favourably against other optimising search techniques such as neighbourhood search and genetic algorithms, both in terms of optimisation ability and calculation cost.

One of the key features of tabu search is the continuous updating of, and reference to, a flexible memory structure. The precise nature of this structure will vary with implementation yet they are all common in that they allow the consideration of effects from previous optimisation attempts when deciding upon a current path for investigation. Through the use of this memory structure, search elements that are used at each iterative step may be ruled out or penalised according to some defined rules; such restricted elements being labelled as tabu. Importantly this helps prevent a tabu search from becoming drawn to and trapped at a single strongly attracting locally optimising solution; in a basic local neighbourhood search this is the primary weakness.

2.2 Diversification and Intensification

To further enhance the ability to prevent the search becoming trapped at a single strong attractor it is often seen that intensification and diversification techniques are used. The basic objective of a diversification technique is to widen the region of solution space considered during a search, allowing a larger range of solutions to be considered. In order to achieve diversification, the search must be forced into a different area of the solution space than it would otherwise encounter. This is accomplished through perturbation that can either occur throughout the search or during the initialisation process; examples of these being multiple minima diversification and restart diversification respectively. Intensification is applied by first identifying what appears to be a promising region within the solution space and then focusing the search upon this region; such focusing is usually accomplished through the exclusive limitation of available moves to those providing solutions within this region.

2.3 Multiple Minima Search Diversification

An example of perturbation that occurs throughout a search process that was considered and tested within this research is a multiple minima search. The base idea behind this diversification is to perturb the search away from any discovered locally optimising solution and encourage it to find another alternative solution within some local vicinity; accordingly in this work the perturbation is set to occur upon discovery of a locally optimising result.

Within this research perturbation was achieved through reversal of a number of the most recent iterative steps taken towards optimisation; effectively this results in returning the search to a previous state that occurred a set number of iterations ago. It is important to prevent the search from proceeding along the same route to optimisation, which would result in rediscovery of the previous optimising solution. In order to achieve this, the tabu tenures associated with the moves subject to reversal in the perturbation are retained. Thus the search will return to a prior state and with the previous optimisation path being heavily penalised it will be encouraged to explore an alternate route ideally ending in a different solution. Each solution discovered is logged before perturbation with the relative strengths of each compared in order to provide the best solution upon final search termination; the strength of each solution being quantified through use of some suitable objective function.

2.4 Restart Diversification

Restart diversification is implemented by selecting a few of the variable parameters and, upon completion of a full search, forcing them to take infrequently encountered values. These variable parameters would be any part of the solution that may be adjusted using any of the search transformations (or moves) defined for use by the search procedure. For the restart diversification applied within this research, the elements selected for perturbation upon search restart were the starting times of each of the defined shifts used in problem initialisation. In order to identify which of the elements was least used within a previous search a count is maintained for each individual element, recording the amount of times each is used.

The perturbation itself was achieved through adjustment to the shift starting time for each of the shifts defined as least used within the previous search. The starting times were increased by 1 hour, decreased by 1 hour or remained as they were, with an equal possibility of selection; in this way the search will consider a different initial state to that of the previous search and will explore a new area of the potential solution space.

2.5 Intensification Techniques

Application of the most commonly used tabu search intensification technique was not appropriate within this work as it was seen to work counter to the multiple minima diversification approach. Indeed use of this basic intensification was seen to harm the quality of solutions found by essentially forcing the search to favour strong local optima.

An alternative intensification technique was created that was used in order to adjust the initial solution before the tabu search itself is initialised. In order to perturb the initial solution in a constructive manner the intensification considers information drawn from a previously conducted optimising search; effectively then this may be known as a restart intensification. Similar to the data used to inform restart diversification, the restart intensification considers the frequency with which each element was involved with the moves conducted between iterations in a previous search. For intensification each shift defined is regarded as an element and it is noted as having been used for each time that a member of staff is moved from or to it within the staff redistribution search stage. 2 counts are maintained for each element; one for staff moving to the associated shift and the other for staff moving from it. Using these, elements that are the most common donators and receivers of staff can be identified. The search is restarted using the same initial conditions as previously which are then perturbed by consideration of the most commonly used elements from the prior search. In order to achieve such perturbation within this work, the elements are used to identify a limited set of elements available for selection within a preliminary search conducted prior to the main tabu search. This restricted element set is defined to allow only those moves in which staff are transferred from previous common donators to previous common receivers. This preliminary search is a local neighbourhood search that proceeds until a first optimising result is found, at which point the tabu search initiates.

3 APPLICATION OF METHOD TO THE POLICE PROBLEM

3.1 The Optimisation Problem

As with all policing organisations Leicestershire Constabulary employ many different types of staff in order to meet the range of demands placed upon them. Local Police Officers (LPOs) are staff members primarily used in responding to calls for service from the general public. Through government enforced targets emphasis is placed upon response to incidents regarded as high priority; these incidents being cases where there is an immediate danger to life or property. For this research it is assumed a reliable and accurate method for quantification of all incident based demand for LPOs exists.

The problem is defined by Leicestershire Police as the need to find the most optimal way of assigning a set number of staff members in order to best meet quantified demand. In this way suitable optimisation methods would be those capable of resulting in a best fit between staffing and demand. There are several factors within this problem, affecting both the availability of staff and the structure of defined shifts. For ease of practical implementation, aligning with the Working Time Directive, a shift length is fixed at 9 hours for the initial research. With fixed shift durations the search will operate through the starting time and number of staff members assigned to each shift. A total available number of staff is given to be distributed over a full calendar week, this then should encourage the optimising search to draw staff members from expected lower demand days of the week to expected higher demand ones.

3.2 Objective Function

In order to be able to assess the quality of any given solution, with regards to its ability to meet demand, an objective function is used. This objective function quantifies into a single real value the amount of demand that is not immediately met for a given solution of the Police problem being considered. To do this the difference between supply and demand of resources is considered for each individual point of time in turn within the entire time period for the problem. If the demand for resources at any point is greater than the supply then the difference between the two is added to a running score total. If instead the supply of resources is equal to or greater than the demand level then no change is made to the score; at these points in time there is no surplus demand to be added on.

Thus it would be expected that an optimising solution would be one that provides the best fit of resources to the demand as possible; in effect this would be equivalent to minimisation of this sum. However the trial problems have been created so that meeting all demand is not possible as this will force the search to continue attempting optimisation until a stopping criteria is met.

3.3 Trial Problem Definition

In order to test the capability of the constructed search algorithm in optimising allocation of staff around a given demand profile, a series of trial problems were constructed. These problems each consist of the pairing of a staffing roster over a period of time and a demand profile that describes the amount of staff required in order to meet the demands at each point in time over the same period. Demand data used for problem construction in this work was defined by a series of demand levels for each quarter hourly period over the entire time range of interest; this is illustrated within Table 1, where a sample 90 minute period (covered by 6 quarter hourly sections) is shown. Demand values quantify the number of incidents which are currently active and requiring the attendance of a member of Police staff within the associated time segment.

Table 1. Sample trial problem demand values across a 90 minute period.

Time	07:00	07:15	07:30	07:45	08:00	08:15
Demand	5.30	5.22	7.35	7.58	6.14	5.05

Staffing levels are instead constructed through definition of a series of shifts; each individual shift being assigned a starting time, duration in hours and staff amount. A sample staffing roster for a single day of the week, in a career where demand occurs 24 hours a day, is displayed in Table 2. For this study, 5 distinct shift starting times are defined for each day of the week; these particular starting times being chosen to reflect one of the shift definition options considered by Leicestershire Police.

Table 2. An example single day staffing roster.

Start	07:00	09:00	14:00	18:00	22:00
Length	9	8	9	6	9
Staff	3	1	2	4	2

3.4 Implementation

3.4.1 Input Initialization

The first stage of the tabu search used in this research is the input data from the relevant sources, which results in the storage of input data in a series of real and integer value arrays. This input data will record both the values descriptive of demand levels for each time interval and the initial shift composition. It is then important to assess this initial setup in order to obtain a benchmark value recording the original score attributed to the roster; without this both the effects of the search and indeed whether optimisation were actually occurring at all would be unknown.

3.4.2 The Tabu Search Moves

Within this study 2 types of move are considered. The first move type performs adjustment to the staffing level assigned to each of the defined shifts whereas the second seeks to adjust the shift structure itself through variation in shift starting times. It should be noted that neither of these will result in the creation of additional shifts within a staffing roster and as such both the total staffing level and the total number of shifts defined within initial setup will be constraints upon the search.

3.4.3 Initial Optimization and Move 1

The optimising search begins through redistribution of the initially assigned staff amongst the shifts defined using the input staff roster. This is achieved by selecting every possible pairing of available shifts and attempting to move a single staff member from the first of the pair to the second. After each of these trial moves is made, the roster score is recalculated to take account of the change that has been made, this score is logged and the move is reversed. In this way every possible roster that may be reached through a single staff exchange move is encountered and has its objective function score recorded. Thus the entire local neighbourhood surrounding the current solution is investigated in the same way as it would be for a neighbourhood search algorithm. The best of these moves is selected to be carried out as a permanent adjustment to the roster; in this case a best move is taken to be that resulting in most improvement to the current roster score towards optimality. Upon discovery and selection of a most improving move the associated elements are marked as tabu in order to discourage their repeated use. In this work this is accomplished through application of a recency penalty to each element; this penalty was initially variable however trial simulations suggested a fixed value of 5.

3.4.4 Implementation of Move 2

The optimisation process through use of move 1 continues until no further improving moves are available to the search and thus a locally optimising solution has been found; whereupon a second stage of the search initiates that will investigate adjustment to the shift starting times. In a similar way as in the staff redistribution stage the local neighbourhood of the current solution is investigated at every search iteration by conducting a series of moves and noting their relative strengths. The best of these moves is again selected for implementation and the associated elements marked as tabu; the tabu status being implemented again with a recency tenure penalty of 5. With completion of this shift starting time adjustment the final stage of the search is to perform a second staff redistribution stage; this being included to encourage optimal use of the newly defined shifts.

3.4.5 Diversification and Intensification

At this point a set of locally optimising results will have been discovered and the best of them will be proposed as a suitable solution. If multiple minima diversification is chosen to be applied then this is the point at which it will initiate, reversing the previous 5 search iterations whilst retaining tabu tenures. This multiple minima search will continue until the desired number of locally optimising solutions have been found, at which point the search will terminate. Within this research it was decided to continue until 10 such solutions were discovered. At this point restart diversification and intensification may be applied through restarting the search from its initial state and applying either one or both of the processes.

The process of application of tabu search with diversification and intensification technique application will continue until some stopping criteria is reached. Multiple minima search will continue until a sufficient number of solutions have been discovered and the restart diversification and intensification approaches will repeat until a specified number of restarts have occurred; in this work 1000 restarts were used when applying restart techniques. This process is summarised within the flow chart presented in Figure 1.

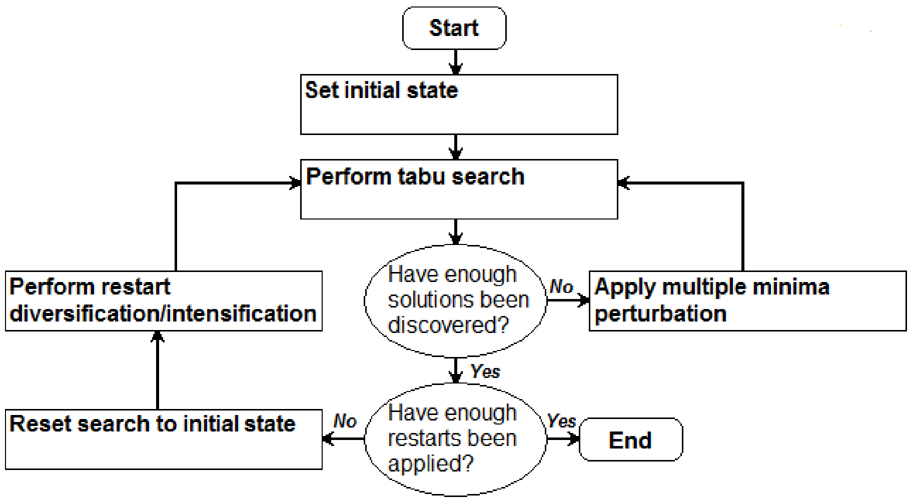

Start
Set initial state
Perform tabu search
Perform restart diversification/intensification
Have enough solutions been discovered?
No
Apply multiple minima perturbation
Yes
Reset search to initial state
No
Have enough restarts been applied?
Yes
End

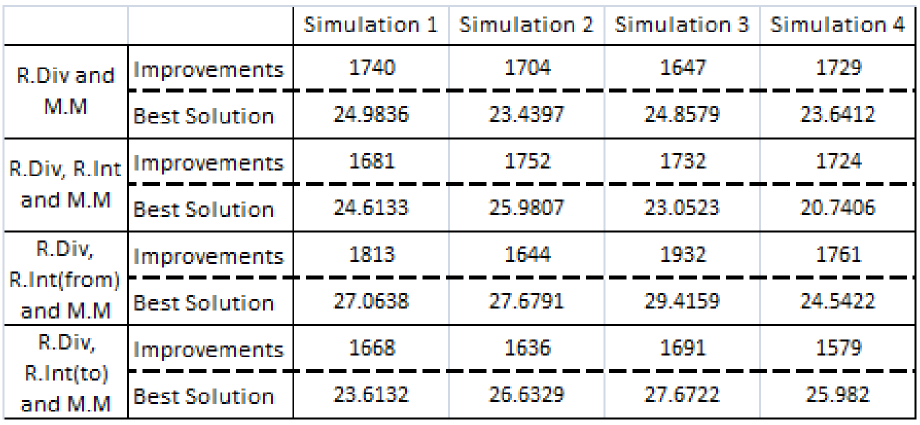
Figure 1. A flow chart describing the structure of the tabu search algorithm used here.

4 RESULTS

A series of simulations were run in order to compare application of different combinations of diversification and intensification techniques when applied to the base tabu search. Each individual simulation entailed performing 10000 tabu searches in sequence with restart diversification and intensification methods implemented between each search where appropriate. The base figure used for comparison in order to show improvement was taken from a single search performed upon a fixed trial problem using the tabu search algorithm with the multiple minima diversification; the resulting score of the objective function used was 39.784. Application of the same tabu search methodology with no tabu tenure implementation is equivalent to performing a steepest descent search and results in an objective function score of 41.071; immediately it is seen that even the basic tabu search method improves upon this value. Results from other tests would then be compared against these benchmark figures in order to indicate any improvements made to solution quality from the search; noted through reduction in this objective value.

Table 3 describes the results from all tests that were performed. In this table, 4 combinations of diversification and intensification approaches are listed. Firstly ‘R.Div and M.M’ refers to simulations in which restart and multiple minima diversifications are applied to the basic tabu search; ‘R.Div, R.Int and M.M’ describing the same combination with the addition of restart intensification. The final two categories are used to signify scenarios in which restart intensification is applied in a less restrictive sense. R.Int(from) indicates that in performing the preliminary search restriction to the available moves is placed only upon the list of shifts that staff may move from; in this way then a reduced list of staff donators is provided however all shifts may be receptors of additional staff in the preliminary search. Similarly R.Int(to) restricts only the shifts that staff may be moved to; so only the most common receptors of staff within the previously conducted search will be allowed as receptors within the ensuing preliminary intensification search. The ‘Improvements’ values indicate in how many of the searches out of 10000 that the application of the associated diversification and intensification techniques offered improvement over the base search with multiple minima diversification. ‘Best Solution’ values describe the score as quantified by the objective function for the most optimising solution found out of all 10000 searches. These `Best Solution’ values are indicative of the quality of the final best solution found by the search (with a lower score identifying a higher quality solution) and indeed allow for comparison of the impact upon final solution quality for each of the various search method combinations.

Table 3. Results from varied tabu search simulations



		Simulation 1	Simulation 2	Simulation 3	Simulation 4
R.Div and M.M	Improvements	1740	1704	1647	1729
	Best Solution	24.9836	23.4397	24.8579	23.6412
R.Div, R.Int and M.M	Improvements	1681	1752	1732	1724
	Best Solution	24.6133	25.9807	23.0523	20.7406
R.Div, R.Int(from) and M.M	Improvements	1813	1644	1932	1761
	Best Solution	27.0638	27.6791	29.4159	24.5422
R.Div, R.Int(to) and M.M	Improvements	1668	1636	1691	1579
	Best Solution	23.6132	26.6329	27.6722	25.982

Each of the combinations of diversification and intensification techniques is shown to provide an improvement to solution quality when applied to the basic tabu search. The inclusion of multiple minima (M.M) and restart diversification (R.Div) approaches results in an average final solution quality of 24.2306 over the 4 trial simulations; this is a notable improvement from the benchmark figure of 39.784, showing an average reduction of 39.1%. Further addition of the full restart intensification process gives an average solution score of 23.5967, equivalent to a score reduction of 40.7% from the benchmark. These two results, with and without restart intensification, are similar to each other and it is suggestive that application of restart intensification has not been of benefit within this work. Indeed both restricted variants of the intensification technique are seen to demonstrate a detrimental effect upon the average solution quality; solution quality dropping by an average of 15.6% and 10.9% for the individual restricted techniques. Figure 2 illustrates the initial roster and demand profile alongside the best found staffing roster from the optimising tabu search techniques of restart diversification and multiple minima diversification. The shaded area represents the quantified demand over time, the thinner line demonstrates the amount of staff provided over time using the initial staffing roster and the thicker line showing the staffing over time given by the best found roster from tabu search application. This shows a good fit between the optimised staffing roster and the demand, demonstrating the effectiveness of the tabu search when applied to this problem.

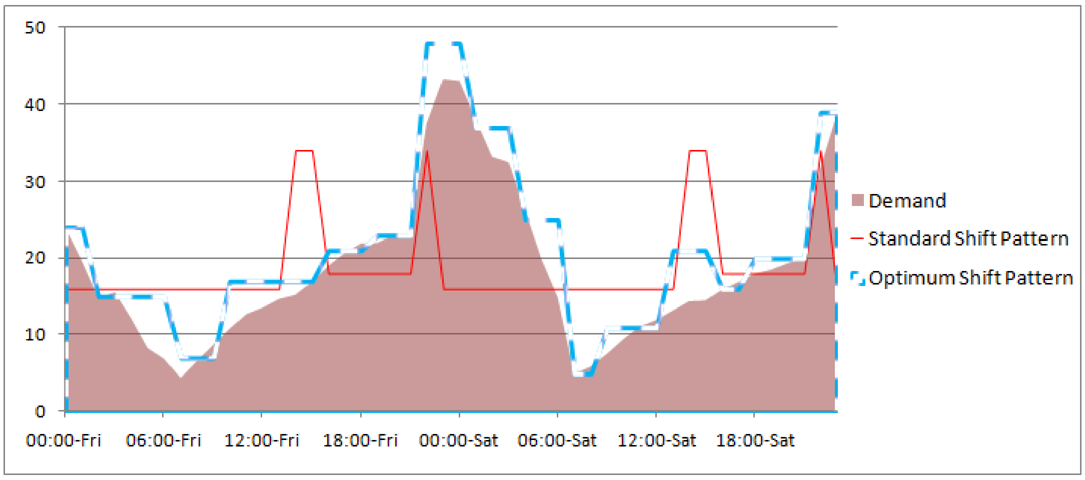


Figure 2. Comparison of initial staff roster against optimised roster from use of tabu search.

Similar quantities of improving searches are seen across all tabu search enhancement combinations and all simulation runs. From this it may be stated that any factor common between all combinations is the controlling factor of the success rate for the search; in this case the common factors are the multiple minima and restart diversification approaches. The rate of improvement is however fairly low, with better solution quality provided in only 17% of the conducted searches. This is suggestive that the basic roster defined is a good initialisation point for the tabu search, with many of the perturbations through restart diversification leading to points from which only lower quality solutions are found.

5 CONCLUSIONS

Tabu search has been shown to have a strong relevance to application on the type of problem considered by this work, producing high quality results with relatively low calculation cost. The multiple minima diversification technique was demonstrated capable of improving results obtained from the basic tabu search by forcing a larger number of potential optimising solutions to be considered in each search. Similarly, restart diversification was seen to provide improvement to the quality of final optimising solutions. This showing not only that restart diversification provides a benefit to the strength of the tabu search, but also that the initial conditions are themselves an important factor to final solutions obtained from this type of problem. The restart intensification procedure offers no improvement to the best solutions provided by a tabu search; both when performed alone and when used in conjunction with the diversification methods. This is seen through the fact that inclusion of this feature within the search leads to only negligible improvement to the quality of final solutions provided, with an average improvement of only 2.6%. The most powerful tabu search method created through compilation of these considerations was tested on a larger range of 15 trial problems representative of the smaller policing regions within Leicestershire. Across these problems an average improvement of 38.2% was seen for final solution quality beyond that provided by steepest descent.

The potential impact that application of tabu search based optimisation techniques may have within the Police and other service industries is notable. The sample staffing roster used to form the base trial problem within this work is an approximation to a realistic roster. Thus comparing the objective function scores of this and the optimised rosters can give an indication of the scale of improvement that may be found in real world problems. The base roster has an objective function value of 386.3274, which compared to the best found result of 20.7406 equates to a difference of 365.5868 person hours of work; using an hourly rate of £13.19 this is equivalent to a weekly effective cost saving of £4822.09. Note that the same total number of staff members are employed in this example, the effective cost saving is derived improvements in the efficiency with which they are used.

Only a limited variety of moves are available within the tabu search defined. Through definition of additional moves it would be possible to increase the flexibility and diversity of the search; this would allow a wider investigation of the solution space improving both solution quality and the chance of restart diversification leading to further improvement. There is further potential to develop the work presented here through implementation of operational research techniques in allowing the variation of system parameters. The authors have conducted work investigating variation of shift structures (through shift duration perturbation), adjustment to total staffing availability and an alternative objective function that seeks to find the most optimal staffing allocation across the most optimal shift pattern.

ACKNOWLEDGEMENTS

The authors would like to acknowledge the assistance provided by Leicestershire Constabulary throughout this research. Notably the assistance of Ch. Supt. Jason Masters, Sgt. Dave Hall and Mrs Carolyn Steptoe in managing Police aspects of the work and of Mr Jonathan White in provision of data.

REFERENCES

Baker K R (1976). Workforce Allocation in Cyclical Scheduling Problems: A Survey. Operational Research Quarterly. Vol. **27** No. 1, pp. 155-167

Burke E K, Kendall G and Soubeiga E (2003). A Tabu-Search Hyperheuristic for Timetabling and Rostering. Journal of Heuristics Vol. **9**, pp. 451-470.

Gaspero L and Schaerf A (2001). Tabu Search Techniques for Examination Timetabling. Practice and theory of Automated Timetabling **III**, pp. 104-117.

Gendreau M, Hertz A and Laporte G (1994). A Tabu Search Heuristic for the Vehicle Routing Problem. Management Science Vol. **40** No. 10 (Oct. 1994), pp. 1276-1290.

Glover F (1994). Tabu search for nonlinear and parametric optimization. Discrete Applied Mathematics Vol. **49** (Mar. 1994), pp. 231-255.

Glover F and Laguna M (1997). Tabu Search. Kluwer Academic Publishers, London.

Hertz A and Werra D (1987). Using tabu search techniques for graph coloring. Computing. Vol. **39** No. 4, pp. 345-351.

Hertz A and Werra D (1990). The tabu search metaheuristic: How we used it. Annals of Mathematics and Artificial Intelligence. Vol. **1** No. 1-4, pp. 111-121.

Jennings O B, Mandelbaum A, Massey W A and Whitt W (1996). Server Staffing To Meet Time-Varying Demand. Management Science. Vol. 42 No. 10 (Oct. 1996) pp. 1383-1394.

Gaspero, L. D. Gartner, J. Kortsarz, G. Musliu, N. Schaerf, A. & Slany, W. The Minimum Shift Design Problem. Annals of Operational Research. Vol. 155 (2007) pp. 79-105.

Burke, E.K. Causmaecker, P.D. Berghe, G.V. & Landeghem, H.V The State of the Art of Nurse Rostering. Journal of Scheduling Vol. 7 (2004) pp. 441-499.

Burke, E.K. Causmaecker, P.D. & Berghe, G.V. A Hybrid Tabu Search Algorithm for the Nurse Rostering Problem. Lecture Notes in Computer Science Vol. 1585 (1999) pp. 187-194.

Musliu, N. Schaerf, A. & Slany, W. Local Search for Shift Desgin. European Journal of Operational Research. Vol. 153 (2004) pp. 51-64.

Ernst, A. T. Jiang, H. Krishnamoorthy, M. & Sier, D. Staff scheduling and rostering: A review of applications, methods and models. European Journal of Operational Research Vol 153 (2004) pp. 3-27.

Naudin, E. Chan, P. Y. C. Hiroux, M. Zemmouri, T. & Weil, G. Analysis of three mathematical models of the Staff Rostering Problem. Journal of Scheduling (2009)