Computers & Operations Research 75 (2016) 150–162

Contents lists available at ScienceDirect

Computers & Operations Research

journal homepage: www.elsevier.com/locate/caor

The multiple team formation problem using sociometry

Jimmy H. Gutiérrez [d], César A. Astudillo [b], Pablo Ballesteros-Pérez [c,*], Daniel Mora-Melià [d], Alfredo Candia-Véjar [a]

[a] Departamento de Ingeniería Industrial, Facultad de Ingeniería, Universidad de Talca, Chile
[b] Departamento de Ciencias de la Computación, Facultad de Ingeniería, Universidad de Talca, Chile
[c] School of Construction Management and Engineering, Whiteknights, Reading RG6 6AW, United Kingdom
[d] Departamento de Ingeniería y Gestión de la Construcción, Facultad de Ingeniería, Universidad de Talca, Chile

ARTICLE INFO

Article history:
Received 15 July 2015
Received in revised form
9 March 2016
Accepted 19 May 2016
Available online 22 May 2016

Keywords:
Multiple team formation problem
Sociometry
Heuristics

ABSTRACT

The Team Formation problem (TFP) has become a well-known problem in the OR literature over the last few years. In this problem, the allocation of multiple individuals that match a required set of skills as a group must be chosen to maximise one or several social positive attributes.

Specifically, the aim of the current research is two-fold. First, two new dimensions of the TFP are added by considering multiple projects and fractions of people's dedication. This new problem is named the Multiple Team Formation Problem (MTFP).

Second, an optimisation model consisting in a quadratic objective function, linear constraints and integer variables is proposed for the problem. The optimisation model is solved by three algorithms: a Constraint Programming approach provided by a commercial solver, a Local Search heuristic and a Variable Neighbourhood Search metaheuristic. These three algorithms constitute the first attempt to solve the MTFP, being a variable neighbourhood local search metaheuristic the most efficient in almost all cases.

Applications of this problem commonly appear in real-life situations, particularly with the current and ongoing development of social network analysis. Therefore, this work opens multiple paths for future research.

1. Introduction

Currently, most private companies are forced to carry out multiple projects in a simultaneous fashion to increase manufacturing capacity and/or to meet market requirements [26], a multi-task paradigm that does not differ historically from the scope of public institutions.

In this context, it is absolutely necessary to make the most of available resources, especially in highly competitive industries in which mark-ups are strongly dependent on how effectively the project was managed [9,11].

Generally, one of the most relevant operational expenses within multidisciplinary projects corresponds to personnel hiring and training [42], and therefore, it seems logical that companies worldwide put significant efforts to ensure that these resources are used properly. However, depending on their combination and interactions as well as task assignment and motivation, project outcomes can vary significantly [14], and in this context, social relationships among the project members take on an important role [8].

In the past, companies usually assigned narrowly focused tasks to single individuals to manufacture a product or carry out a service, whereas a supervisor or foreman was tasked with watching over the process outcomes [41]. In that environment, worker allocation was not a problem because human interactions were minimised. However, according to Bailey [7], this strategy has been replaced by a scheme in which groups of people who are generally organised into work cells develop important components of the project using higher levels of social and skills interaction. Later, after a process of integration, these components are assembled to create the final product.

Thus, it is obvious that social interaction plays an important role in project success, but paradoxically, most research has been carried out at the single individual level rather than at the team level [14] because the latter is far newer and likely still removed from daily and real-life applications [32].

Nevertheless, a few pieces of research have been published that prove team productivity is strongly influenced by the health of the social relationships among group members, i.e., cohesion or dissociation [8], although these relationships have been rather

* Corresponding author.
E-mail address: p.ballesteros@reading.ac.uk (P. Ballesteros-Pérez).

http://dx.doi.org/10.1016/j.cor.2016.05.012

Table 1
Basic bibliography of the team formation problem and summary of contributions (Part 1).

Ref.	Year	Specific incremental contributions
Lappas et al. [37]	2009	First computation of a solution to the problem of creating a team of experts using social networks.
Dorn and Dustdar [22]	2010	Adds an heuristic procedure that handles selected concepts relative to expert interaction network extraction and skill profile creation before proceeding to team formation itself.
Li and Shan [38]	2010	Generalises the Team Formation Problem by associating each required skill with a specific number of experts. In addition, this work extends the Enhanced-Steiner algorithm to a generalised version for generalised tasks, devises a density-based measure to improve the effectiveness of the team, and finally, presents a novel grouping-based method that condenses the expertise information to a group graph according to the required skills.
Yin et al. [54]	2011	This is the first work to study diversity in the social graph and facilitate its effect in the Expert Group Formation.Particularly, this approach states that the team members must not only meet the skill requirements of the task but also must be diversified according to a metric based on social influence.
Farhadi et al. [24]	2011	This work adds skill levels or grading for the tasks/projects to be performed while seeking the minimum communication cost among team members.
Sorkhi et al. [49]	2012	This work proposes a game theoretic framework to find and rank top-k teams that meet given skillset requirements.
Farhadi et al. [25]	2012	This work determines the skill level of each expert based on his/her skill/s and collaboration among neighbours, and a graph is aggregated to the set of skilled expert groups. Additionally, the RarestFirst algorithm is extended to a more generalised version, and the communication cost definition is customised to improve team efficiency.

neglected thus far because they were considered too difficult to measure and/or to include in mathematical models [26].

The problem of allocating multiple human resources to multiple projects involves the distribution of different people with various skillsets to a series of projects that usually require more than a single area of expertise while optimising other pre-established criteria [31]. This problem has drawn much attention in the scientific literature thus far (e.g., [26,14,8,40]), but most of it stems from the psychological and behavioural perspective.

However, dating back to 1941, techniques have existed that study the structure of groups through the web of interpersonal relationships that occur within it Moreno [45]. This technique is known as “sociometry” and constitutes one of the few quantitative tools used for describing a group's health as well as its individuals' social status.

Contributions and paper outline. This research proposes the Multiple Team Formation Problem (MTFP) as a mathematical programming model for maximising the efficiency understood as the number of positive interpersonal relationships among people who share a multidisciplinary work cell. To this end, the “sociometric matrix”, one of the main tools for psychological analysis devised by Moreno [45], is used as the main input of the problem because it provides an excellent quantitative vision of how each potential group member perceives and is perceived by his/her fellow peers within a multidisciplinary group.

Three algorithms are proposed for solving the problem. A Constraint Programming approach (CP) implemented in a IBM ILOG CP Optimised solver, a Local Search heuristic (LS) and a Variable Neighbourhood Search metaheuristic (VNS), both implemented in Java. The experiments over three classes of instances show that it is possible to identify different structures of the problem with significantly different levels of efficiency.

The remainder of the paper is organised as follows. Section 2 opens with a review and classification of the recent literature on the Team Formation problem, as it has been commonly referred to in the Operational Research context for single-group situations. In Section 3, the proposed optimisation model and a decomposition property are presented. Section 4 illustrates the algorithms used for solving the problem: CP, LS and VNS. The results of the computational experiments are analysed in Section 5. Finally, Section 6 presents some conclusions and suggests future work.

2. Literature review

The success or failure of projects is directly related to the individual talent of the participants and how they are assigned to their respective tasks. However, currently, few models are used by companies for people-task assignment [32], and this is mostly the case because scientific models are generally not intuitive and/or operatively difficult to implement for daily-basis operations. Analogously, human relationships and interactions are difficult to incorporate in mathematical models because little information is generally known about their quantification [26] and how data must be gathered [8].

However, the grouping of individuals for effective collaboration is not new. Recurrent attempts are found in psychology and/or anthropology in which several tests (e.g., Myers-Briggs) or metrics (e.g., Kolbe Conative Index) [26] are used to measure people's personalities and forecast their future performance within pre-existing groups.

Similarly, previous studies have proven that sociometric analysis constitutes a valuable tool for predicting diverse performance criteria (e.g., productivity, combat effectiveness, training ability, leadership) [6]. Nevertheless, the evolution of the TFP with consideration of social network analysis is still relatively limited.

In the literature, the first attempt to combine functional (skills) requirements with a network structure dates back to only 2004 [28]. In this first experimental analysis, the authors studied how different graph structures shared among the individuals affected team performance, although it did not address these findings for grouping people further than as a descriptive tool. It was not until five years later that another study directly tackled the problem of finding a team of experts in social networks [37]. For the first time, this seminal work addressed the challenge of gathering a group of experts from a vast professional network with a required set of skills while maximising their social compatibility using a metric/function known as communication cost.

Following this paper, many other works have been steadily published at a rapid pace, including most of them just small improvements to Lappas et al.'s (2009) original ideas. Most of these works and their contributions are summarised chronologically in Tables 1 and 2.

Similar to many other grouping optimisation problems, (e.g. [23,3,4]), the TFP is classified as NP-Hard (proven by [37] for single team instances). In addition, the vast majority of previous studies differ from our work in three major aspects. First, the TFP works with a single-project whereas in this paper, a multi-project situation is the dominant scenario. Second, personnel were only allocated as full-time resources (divisions of people, i.e., part-time allocations, were not allowed). Third, the pool of human resources applied was a complete social network (e.g., the entire scientific research community based on papers published and co-authored or LinkedIn), unlike in this work, in which a generic and much

Table 2
Basic bibliography of the Team Formation problem and summary of contributions (Part 2).

Ref.	Year	Specific incremental contributions
Gajewar and Sarma [27]	2012	This work presents a 3-approximation algorithm for the single-skill team formulation problem and shows that the same approximation can be extended to a special case of multiple skills. This problem generalises the formulation studied by Lappas et al. [37], which measure team compatibility in terms of diameter or spanning tree.
Farhadi et al. [25]	2012	This work proposes a framework known as TeamFinder that provides a method to aggregate a set of experts that are strongly correlated based on their skills and connections.
Li et al. [39]	2013	This work states that the constructed teams require an adequate number of experts for each required skill. Therefore, this work considers the specific number of experts needed to devise a generalised Enhanced-Steiner algorithm first proposed by Lappas et al. [37] and presents a new grouping-based method that condenses the expertise information.
Shi and Hao [48]	2013	This work formulates the Team Formation problem from the standpoint of multi-criteria decision-making task ranking in social networks.
Teixeira and Huzita [50]	2014	This work presents a context-aware multi-agent mechanism to support human resource allocation in distributed projects. This mechanism performs the people allocation to the tasks of a project by taking into account the participants' contextual information (culture, idiom, temporal distance and previous experience), the requirements of the tasks, and the interpersonal relationships among the human resources.
Agrawal et al. [1]	2014	This work groups classmates into sections such that the overall gain for all student groups is maximised and is the first time that non-overlapping groups are considered; this is considered a "much harder and complex problem".
Awal and Bharadwaj [5]	2014	This work combines the team formation problem with the Collective Intelligence (CI) concept. The CI emerges from members' collaborations and attempts to maximise the potential of the team of experts rather than only aggregating individual potentials. A genetic algorithm-based approach is also applied.

smaller network is used to depict the human resources that might be found within the same company or classroom, as an example. As a result of the former, the MTFP becomes a quadratic optimisation problem, stressing the need for implementing metaheuristic techniques such as the ones discussed later.

Considering these major differences between previous studies and our work, only a handful of rather recent papers are conceptually similar. The first group is composed of a series of papers focused on combining Sociometry and Genetic Algorithms to solve the problem of grouping people, i.e., mostly classmates, based on social relationships [18,17,2].

These papers make use of a composite objective function based on sociometry metrics, which is somewhat similar to the approach that is implemented in this work, and also accounts for the possibility of forming different and simultaneous groups. The main differences are that these groups must be equal in size (number of members), people's skills are neglected, and people themselves cannot be allocated as part-time.

The second group of papers [8,10] has already integrated the assignment of multiple human resources to multiple projects, thus allowing for part-time allocations (although up to 50% dedications only and no smaller fractions) and taking into account the sociometric approach. However, these papers do not solve the problem computationally and only propose a manual heuristic method. Therefore, these last papers are considered to actually formulate the problem in rudimentary terms similar to the ones described in this work, but they do not provide a solution, likely due to the reality of its inherent high complexity.

Finally, other ramifications of this topic cover how to obtain information on the social network itself (e.g., filling out the sociometric matrix) given a pool of human resources with different skills [16,52], especially if the assumption is that a discrepancy usually exists between official and private human behaviour [34]. In this manner, by including other similar but closely related problems, an even larger volume of literature can be found in Operations Research related to the need to match groups of professionals with projects that demand a known set of skills while optimising other variables, i.e., number of people involved, economic profits derived from task completions, execution time, etc. Computationally, these problems have been traditionally solved using such techniques as integer linear programming [29], mixed integer programming [26], simulated annealing [12], branch-and-cut algorithms [55], tabu search [3], artificial bee colony [21] and genetic algorithms [52]. However, this formulation, also known as the "General Assignment Problem", does not take into account social structures among individuals, and thus, it will not be further considered and be left for future research.

3. The Multiple Team Formation Problem (MTFP)

In this section, the basic notation and problem definition, an example instance, an optimisation model and its decomposition property into feasible solutions space for MTFP are presented.

3.1. Notation and problem definition

Notation for sets and parameters are given as follows:

- $\mathcal{P}$ Set of projects to which people are allocated. Then $\mathcal{P} = \{p_1, p_2, \ldots, p_m\}$.
- $\mathcal{H}$ Set of people available that can be allocated to $\mathcal{P}$. Then $\mathcal{H} = \{h_1, h_2, \ldots, h_n\}$. Note that each person is considered to have a single skill only.
- $\mathcal{K}$ Set of skills or areas of expertise that people in $\mathcal{H}$ possess, all of which are required by at least one project in $\mathcal{P}$. Then $\mathcal{K} = \{k_1, k_2, \ldots, k_f\}$.
- $\mathcal{Q}$ Set of lists of people $\{Q_1, Q_2, \ldots, Q_f\}$ available in $\mathcal{H}$ who share the same skill $k \in \mathcal{K}$.
- $\mathcal{D}$ Set of people's allowed time allocation fractions. Then $\mathcal{D} = \{d_1, d_2, \ldots, d_t\}$. For example, $\{0, 1\}$ (full-time allocation) and $\{0, 0.5, 1\}$ (half-time allocations).
- R Project Requirements matrix that specifies how many people (or fractions of people, if allowed) with the same skill k_a are needed for each particular project $p_l \in \mathcal{P}$. This matrix has dimensions $f \times m$, i.e., the total number of skills by the total number of projects, whereas each element r_{al} is always a non-negative real number.
- S Sociometric matrix with elements denoted as s_{ij}. This is a non-symmetrical square matrix with size $n \times n$ (total number of people available) that contains the predisposition of each individual $h_i \in \mathcal{H}$ for working with individual $h_j \in \mathcal{H}$. These matrix elements can take on three possible values, $s_{ij} = \{-1, 0, +1\}$, where value $+1$ is chosen if h_i is willing to work with h_j, value -1 if h_i would prefer not to work with h_j, and 0 if h_i is neutral to work with h_j (either

because h_i is undecided about h_j or because h_i does not know h_j). Furthermore, it is obvious that the diagonal elements s_{ii} are always equal to $+1$.

W List of m numbers $\{w_1, w_2, \ldots, w_m\}$ that describes the weight or priority of the projects. For the sake of maintaining the objective function in the range $[0, 1]$ the sum of all elements equals 1.

The *Project Efficiency* is defined by the sum of evaluations of relationships over all the pairs of people working on the same project. Hence, the project efficiency of the project l (e_l) is given by:

$$e_l = \frac{1}{2}\left(1 + \frac{\sum_{h_i,h_j \in H} s_{ij} x_{il} x_{jl}}{[\sum_{k_a \in K} r_{al}]^2}\right) \forall\, p_l \in \mathcal{P}$$

where the first $\frac{1}{2}$ and 1 terms allow shifting the interval of variation from $[-1, 1]$ to $[0, 1]$. $X = [x_{il}]$ is the set of decision variables modelling the allocation matrix, for which each element $x_{il} \in \mathcal{D}$ specifies the time fraction for which each person available $h_i \in \mathcal{H}$ is allocated to each project $p_l \in \mathcal{P}$. Matrix X has dimensions $n \times m$.

In e_l, the numerator in the fraction of the formula, represents the total relationships between each pair h_i and h_j weighted by the total time occupied by them. This *efficiency* measure is calculated respect to the maximum possible efficiency for this project, which coincides with the numerator expression when all $s_{ij} = +1 \;\forall\, h_i, h_j \in \mathcal{H}$ resulting in the denominator.

Definition. Given a set of n available people each of whom are categorized by having just a single skill $k_d \in \mathcal{K}$ and given a set of m projects that require a specified amount of people per skill, the MTFP can be understood as the problem of maximising the "Global Efficiency", E, corresponding to the sum of project efficiencies weighted by w_l values. Therefore:

$$E = \sum_{p_l \in \mathcal{P}} w_l e_l$$

Note: The Global Efficiency is a convex linear combination of e_l so that: $0 \le E \le 1$.

Fig. 1 shows the main components of the problem and also illustrates an example of a feasible solution.

3.2. An example of the MTFP

For the sake of clarity, Tables 3 and 4 show the data associated to one example instance of the MTFP with 5 projects and 50 people who are categorized into 5 skills. In this case, the set of people's allowed time allocation fractions was $(0.0; 0.25; 0.5; 0.75; 1.0)$. Table 5 shows an optimal solution for the instance.

The way of ensuring the existence of an optimal solution was achieved by designing the problem backwards. Several random people with different skills were selected beforehand to be allocated to five projects. The predisposition to work with the coworkers from their own project was set always as $s_{ij} = +1$, whereas the rest of people had random predispositions $s_{ij} = \{-1, 0, +1\}$. Therefore, the existence of a solution with $E = 1$ was, not only

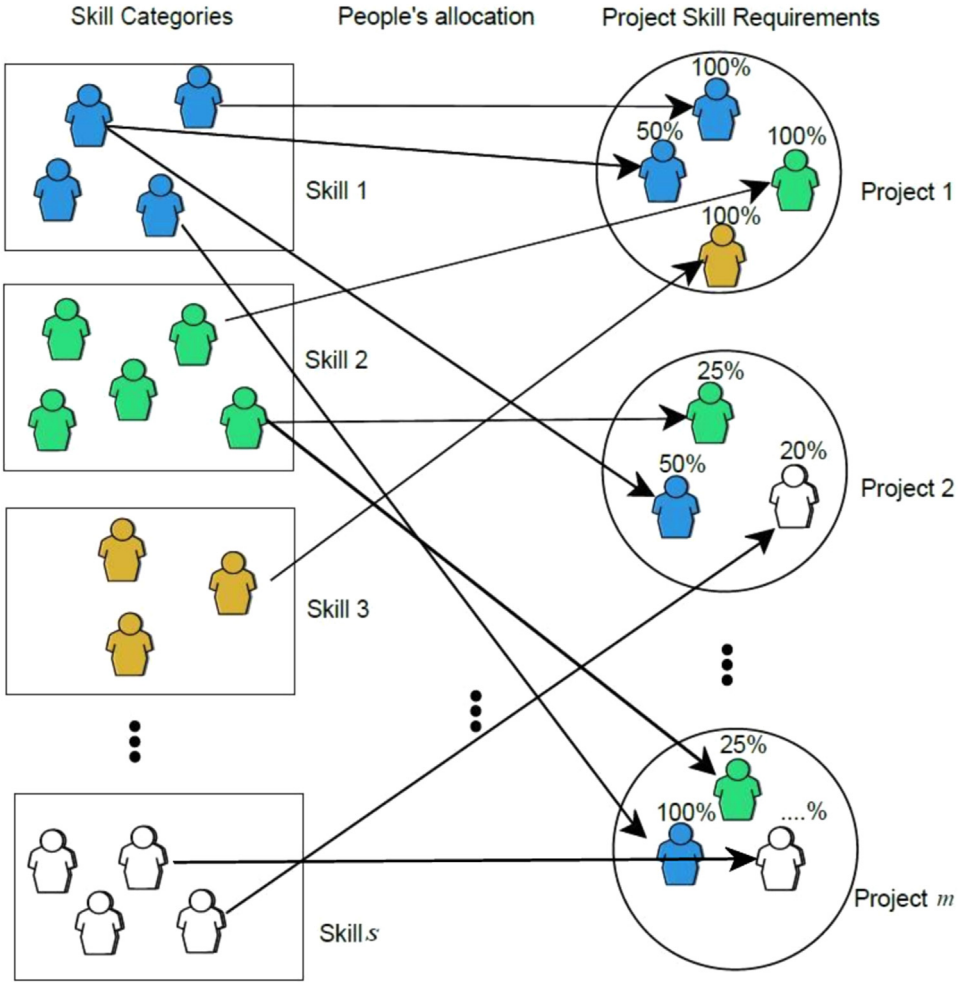


Fig. 1. Multiple team formation problem scheme.

Table 3
Requirements matrix example.

Skill (k_a)	**Project** (p_l)				
	1	**2**	**3**	**4**	**5**
1	2.50	0.50	1.75	1.50	1.00
2	2.50	2.00	3.50	2.25	0.75
3	2.75	1.00	1.50	1.75	3.75
4	1.50	2.00	0.25	0.50	2.00
5	1.25	1.25	1.50	2.25	1.00
$\sum_a r_{al}$	10.50	6.75	8.50	8.25	8.50
Weights (w_l)	0.247	0.159	0.200	0.194	0.200

certain, but the specific optimal allocation of people known. With an analogous procedure for creating problems, the algorithms proposed later can also be tested in problems where a perfect Global Efficiency is achievable, unlike in real-life settings where an optimum solution, if exists, will not be known.

3.3. Discrete optimisation formulation for the MTFP

A discrete quadratic optimisation model is presented for the MTFP. In this first formulation, the variables representing people's allocations (x_{il}) are naturally non-integer. After a simple transformation, these variables become integers. A description of former mathematical model is presented below, whereas the latter is explained afterwards.

Maximise

$$E = \sum_{p_l \in \mathcal{P}} w_l e_l \tag{1}$$

Subject to

$$\sum_{p_j \in \mathcal{P}} x_{ij} \leq 1 \quad \forall\, h_i \in \mathcal{H} \tag{2}$$

$$\sum_{h_i \in Q_a} x_{il} = r_{al} \quad \forall\, p_l \in \mathcal{P}, \quad \forall\, k_a \in \mathcal{K} \tag{3}$$

$$x_{ij} \in \mathcal{D} \quad \forall\, h_i, \quad h_j \in \mathcal{H} \tag{4}$$

Table 5
Optimal allocation matrix for the example instance.

People (h_i)	**Skill** (k_a)	**Project** (p_l)					**Allocation**
		1	**2**	**3**	**4**	**5**	
1	1	0.25	0.50	0.25	0.00	0.00	1.00
2		0.25	0.00	0.25	0.00	0.50	1.00
⋮		⋮	⋮	⋮	⋮	⋮	⋮
8		0.50	0.00	0.00	0.00	0.00	0.50
9	2	0.00	0.00	0.50	0.25	0.00	0.75
10		0.25	0.25	0.25	0.25	0.00	1.00
⋮		⋮	⋮	⋮	⋮	⋮	⋮
22		0.00	0.25	0.00	0.00	0.00	0.25
23	3	0.25	0.00	0.00	0.00	0.75	1.00
24		0.00	0.50	0.00	0.25	0.25	1.00
⋮		⋮	⋮	⋮	⋮	⋮	⋮
34		0.50	0.50	0.00	0.00	0.00	1.00
35	4	0.00	0.50	0.00	0.00	0.00	0.50
36		0.00	0.25	0.00	0.00	0.50	0.75
⋮		⋮	⋮	⋮	⋮	⋮	⋮
44		0.00	0.00	0.50	0.25	0.00	0.75
45	5	0.25	0.25	0.00	0.25	0.25	1.00
46		0.25	0.25	0.50	0.00	0.00	1.00
⋮		⋮	⋮	⋮	⋮	⋮	⋮
50		0.50	0.00	0.00	0.25	0.00	0.75
	Weights (w_l)	0.247	0.159	0.200	0.194	0.200	
	Efficiencies (e_l)	1.000	1.000	1.000	1.000	1.000	
	$w_l \times e_l$	0.247	0.159	0.200	0.194	0.200	
		$E = \sum e_l \cdot w_l$				**1.00**	

This model seeks to maximise the *Global Efficiency*(E) represented by the objective function (1). In this equation each Project Efficiency (e_l) detailed in (3.1) contains a non-linear component in the numerator that quantifies the perception of one individual about another taking into account the time that both could work together. Each person is not allowed to work more than 100% of his/her working time (2). For each project the requirements over all skills must be exactly met (3). Constraint (4) defines the rule domain of decision variables.

The decision variable x_{ij} is defined as a discrete set $\mathcal{D}$, where each element is a rational number as noted above. Therefore, the optimisation problem is a quadratic discrete programming problem with non-integer variables. For simplicity it is assumed that

Table 4
Sociometric matrix example.

People (h_i)	→	**1**	**2**	**3**	⋯	**8**	**9**	⋯	**22**	**23**	⋯	**34**	**35**	⋯	**44**	**45**	⋯	**50**
↓	**Skill** (k_a)	1					2			3			4			5		
1	1	1	−1	1	⋯	1	−1	⋯	1	1	⋯	0	1	⋯	0	1	⋯	−1
2		1	1	−1	⋯	1	0	⋯	1	1	⋯	1	−1	⋯	0	1	⋯	0
3		−1	1	1	⋯	1	1	⋯	1	−1	⋯	1	−1	⋯	0	1	⋯	−1
⋮		⋮	⋮	⋮	⋱	⋮	⋮	⋱	⋮	⋮	⋱	⋮	⋮	⋱	⋮	⋮	⋱	⋮
8		−1	1	0	⋯	1	1	⋯	0	0	⋯	−1	1	⋯	1	−1	⋯	1
9	2	1	1	−1	⋯	1	1	⋯	1	1	⋯	−1	1	⋯	1	−1	⋯	0
⋮		⋮	⋮	⋮	⋱	⋮	⋮	⋱	⋮	⋮	⋱	⋮	⋮	⋱	⋮	⋮	⋱	⋮
22		−1	0	−1	⋯	1	1	⋯	1	1	⋯	1	1	⋯	1	1	⋯	1
23	2	0	0	1	⋯	1	1	⋯	−1	1	⋯	0	1	⋯	1	−1	⋯	1
⋮		⋮	⋮	⋮	⋱	⋮	⋮	⋱	⋮	⋮	⋱	⋮	⋮	⋱	⋮	⋮	⋱	⋮
34		−1	0	0	⋯	1	−1	⋯	1	1	⋯	1	1	⋯	1	1	⋯	−1
35	2	0	0	0	⋯	1	1	⋯	−1	−1	⋯	−1	1	⋯	1	1	⋯	0
⋮		⋮	⋮	⋮	⋱	⋮	⋮	⋱	⋮	⋮	⋱	⋮	⋮	⋱	⋮	⋮	⋱	⋮
44		−1	1	−1	⋯	1	1	⋯	−1	1	⋯	0	1	⋯	1	1	⋯	1
45	2	0	−1	0	⋯	−1	1	⋯	−1	1	⋯	0	1	⋯	1	1	⋯	1
⋮		⋮	⋮	⋮	⋱	⋮	⋮	⋱	⋮	⋮	⋱	⋮	⋮	⋱	⋮	⋮	⋱	⋮
50		1	−1	0	⋯	1	1	⋯	1	−1	⋯	1	1	⋯	−1	0	⋯	1

the equally spaced elements of $\mathcal{D}$ form an arithmetic progression with constant "α" and then each element $d_u \in \mathcal{D}$ could be re-defined as $d_u = (u-1) \times \alpha \;\; \forall\, d_u \in \mathcal{D}$, where α is non-negative rational number between 1 and 0. Thus it is possible to shift the domain of x_{ij} ($x := \frac{x}{\alpha}$) to the integer interval $[0, \frac{1}{\alpha}]$ and rewrite the mathematical model as:

$$e_l = \frac{1}{2}\left(1 + \frac{\sum_{h_i,h_j \in H} S_{ij} x_{il} x_{jl} \alpha^2}{[\sum_{k_a \in K} r_{al}]^2}\right) \forall\, p_l \in \mathcal{P} \tag{5}$$

Maximise

$$E = \sum_{p_l \in \mathcal{P}} e_l w_l \tag{6}$$

Subject to:

$$\sum_{p_j \in \mathcal{P}} x_{ij} \leq \frac{1}{\alpha} \;\; \forall\, h_i \in \mathcal{H} \tag{7}$$

$$\sum_{h_i \in Q_a} x_{il} = \frac{r_{al}}{\alpha} \;\; \forall\, p_l \in \mathcal{P}, \quad \forall\, k_a \in \mathcal{K} \tag{8}$$

$$x_{ij} \; integer \;\; \forall\, h_i, h_j \in \mathcal{H} \tag{9}$$

3.4. Decomposition property of the feasible solutions space

The problem of allocating a subset of n people available for the m projects while meeting each project people requirements can be broken down into f subproblems which consist of allocating each set of people Q_a within the same skill $k_a \in \mathcal{K}$ while fulfilling each project requirements for that specific skill.

Let ϕ^a be the feasible solution space for the skill k_a sub-problem in Q_a. Therefore, a simple property of the feasible solutions space ϕ for the MTFP is that any feasible solution can be considered as the union of the different sub-solutions for each skill.

This decomposition property, which only holds for situations in which people are considered to have a single skill, suggests an algorithmic approach for constructing feasible solutions that will be used by the heuristics proposed in the following section. Conversely, in situations where people can have multiple skills, each solution needs to be built up at once from scratch since it cannot be divided into interchangeable feasible sub-solutions. This happens because the same person might be involved in different skill groups, but is equally unable to work more than 100% of his/her time. Therefore, the alternative of multi-skilled people is not considered at this first multi-project treatment since it involves a significantly higher level of complexity.

4. Algorithms proposed for solving the MTFP

In this section three algorithms for solving the MTFP are proposed: Constraint Programming, Local Search and Variable Neighborhood Search. Additionally, this section also details a procedure for constructing feasible solutions.

In the first part of the section, we describe all the basic concepts and relevant bibliography about the above mentioned algorithms, and subsequently, we present a discussion about how these algorithms are applied to the MTFP.

Constraint Programming (CP) is a programming paradigm that is helpful for finding solutions to combinatorial optimisation problems. CP is based primarily on computer science fundamentals, such as logic programming and graph theory, as opposed to other programming paradigms, e.g., mathematical programming which is based on numerical linear algebra. CP is based on a more general computational problem known as Constraint Satisfaction Problem (CSP), designed for solving feasibility problems. Particularly, in CP, an optimisation problem is solved as a series of feasibility problems, computing a bound for the objective value at each iteration. This process is repeated until achieving a proof of optimality. More details about the history and main concepts of CP can be found in Pesant [46].

A CP model applied to a constrained optimisation problem is expressed by means of decision variables, domains, constraints and objectives that must be minimised (or maximised), which can be mathematical or symbolic. CP first uses a filtering procedure for reducing values from a variable domain, discarding those alternatives that are not possible within the feasible space. This information is propagated through the constraints, allowing progressive reductions of the variable domains. Typically the search space is explored by the construction of a search tree in a manner analogous to a branch and bound scheme; the objective function is managed as a bound constraint such that if a new improving solution is found a new bound is introduced. In this process non-improving assignments are infeasible. One of the biggest strengths of CP is the mechanism that explores the search space of partial assignments, producing feasible solutions.

As stated in Russell and Norvig [47], if an existing CSP solving system is available, it is frequently more efficient to solve the problem using it rather than implementing a custom solution. Furthermore, CSP solver systems can be faster when compared to other type of solutions such as state-space searchers, because the former are designed for efficiently eliminating large portions of the search space. Some commercial companies have developed CSP solving systems, that can be used to formulate CP models as well as mathematical programming models. In this sense, CP can be used as a heuristic for finding solutions of mixed integer programmes; in particular, in Section 4.1 we will describe in detail how CP is able to solve a mixed integer programme for the MTFP.

Several applications of CP for solving combinatorial optimisation problems are known. To mention a few applications, Caprara et al. [15] applied CP to solve the railway crew management problem, whereas De Backer et al. [20] successfully applied CP to a vehicle routing problem. More recently, Hooker [33] suggested how an integrated modelling framework can retain, and even enhance, the modelling power of CP while allowing the full computational resources of mathematical programming. The paper of Blum et al. [13] illustrates experiences in hybridising metaheuristics with CP.

Local Search (LS) is an old approach for solving discrete optimisation problems. More specifically, the 2-opt heuristic was applied in 1958 for solving the Traveling Salesman Problem (TSP) [19]. LS is a simple and yet effective method for searching feasible solutions based on an iterative procedure. The structure of a neighborhood is defined in order to search for a better solution in a reduced space so that finding a neighbour of the current solution is fast. Hence, thousands of iterations are normally executed until a specified maximum number is achieved or no better solution is found up to some given iteration. Empirical observations have demonstrated the power of LS for finding good solutions for many discrete optimisation problems. Nonetheless, optimal solutions can only be found for a small class of problems. The expression "trapped in a local optimum" is normally used to illustrate the limitation of the heuristic. LS has proven to be a good heuristic for solving important family of problems like the above-mentioned TSP and a very difficult problem known as Job Shop Scheduling, Van Laarhoven et al. [51]. In spite of its advantages, LS possesses some known drawbacks. Particularly, for many problems, the performance of LS decreases as the size of the problem increases. Many authors have proposed solutions for this situation, designing strategies for escaping from local optima, being one of the most prominent a method called Simulated Annealing [36,35]. An alternative mechanism that inherits the properties of LS is the

so-called Variable neighbourhood Search (VNS) algorithm, proposed by Mladenović and Hansen [43]. Essentially, VNS implies a LS with several types of neighbourhoods, switching from one to another when no improvement in the current solution is found. The assumption is that a local minimum with respect to one neighbourhood function is not necessarily a local minimum when using a distinct neighbourhood. The VNS algorithm as well as some variants and applications are discussed in Hansen et al. [30]. Nowadays, the concept of metaheuristics has emerged as a strong research field that focuses on the study of novel algorithmic approaches that have been particularly effective for solving difficult optimisation problems.

For an updated description of the numerous metaheuristic approaches please refer to the book of Xing and Gao [53]. Among these classical metaheuristics such as Tabu Search, Greedy Randomized Adaptive Search Procedures (GRASP), Genetic Algorithms (GA), etc. it is possible to find examples of traditional approaches. In this sense, techniques based on metaheuristics are numerous and in recent years the development of methodologies to compare the performance of different algorithms has been of interest to the scientific community [44].

4.1. CP – Constraint Programming

As explained in Section 3.4, the decomposition property allows the obtention of a feasible solution for the MTFP by merging the different sub-solutions for each skill. For each skill k_a, the subproblem of the MTFP for k_a consists on allocating each set of people Q_a within the same skill $k_a \in \mathcal{K}$, while at the same time fulfilling the requirements of each project for that specific skill. According to the formulation for the MTFP given in Section 3.3, the sub-problem for k_a is a special case but two considerations must be taken account:

1. The sub-problem is a decision problem (and not an optimisation problem) so the objective function expressed in Eq. (6) should not be considered.
2. The set of constraints expressed in Eq. (8) must be particularised only for the skill k_a. Performing the necessary modifications, the resulting Equation is as follows:

$$\sum_{h_i \in Q_a} x_{il} = \frac{r_{al}}{\alpha} \quad \forall\, p_l \in \mathcal{P} \tag{10}$$

The procedure for constructing feasible solutions for the MTFP uses CP as detailed in Algorithm 1.

Algorithm 1. Search feasible solution for MTFP using CP.

Input: ϕ^a for each $Q_a \in \mathcal{H}$ and $k_a \in \mathcal{K}$.
Output: Feasible Solution for the MTFP.

1: **for each** $Q_a \in \mathcal{H}$ **do**
2: Using CP, obtain a feasible solution in the space ϕ^a with $k_a \in \mathcal{K}$, considering the subproblem formed by constraints (7), (10) and (9)
3: Join the solutions for the f subproblems in step 1, forming a feasible solution for MTFP.

4.2. Local search

Prior to describing the Local Search algorithm for the MTFP, the concept of neighbourhood is required. This neighbourhood will also be used later by the Variable Neighbourhood Search (VNS) algorithm.

4.2.1. The neighbourhood k-skill N^k

Let X be the allocation of skilled people to projects so that the person h_i is assigned to the project p_l with a time dedication d_u (full-time, part-time,…, no time). Then, a feasible solution X could be represented by a set of triples (h_i, p_l, d_u) so that the demand of skilled people by the m projects is satisfied. Let also consider that the people can be clustered according to their skill, belonging each person to a single skill k_a only.

Let X be a feasible solution given by $X = (h_i, p_l, d_u)$. For each $X \in \phi$ (where ϕ is the set of feasible solutions of the problem), ${Q_a}^X$ denotes the set of people with skill k_a allocated to the m projects. It is worth noting that any single-skilled person can be allocated to either no project, several projects or even all projects. Analogously, several people with the same skill can be allocated to a one or several projects at a time.

The neighbourhood "1-Skill", denoted by N^1, is defined in Algorithm 2. The first step consists of selecting one skill at random. Next, the algorithm removes all the assignments (people allocations) related to k_a. Finally, the assignments for skill k_a are reallocated using other specialists with the same skill. This, results in the creation of a new feasible solution X' which is the algorithm output.

Fig. 2 shows an example of neighbourhood N^1. The left part of the figure represents a solution X for the MTFP instance. The right part of the figure shows the corresponding neighbour solution X'. In this example we assume that the skill 1 was randomly selected. This neighbour maintains all the assignments for the skills except for skill 1. In red are shown the new assignment for skill 1 obtained using Algorithm 1.

The time required to obtain a neighbour solution is directly related to step 3 of Algorithm 2, which essentially consists in solving a CP for a single skill. In this case, CP is used for solving the constraint satisfaction subproblem to generate feasible solutions for each skill, which are used by our algorithms to solve the optimisation problem. As observed in our experimental results, CP was able to find neighbour solutions very quickly. From a theoretical perspective, in the worst case scenario, the time spent by CP could be exponential.

Algorithm 2. N^1.

Input: An MTFP instance tuple $\langle \mathcal{P}, \mathcal{H}, \mathcal{K}, \mathcal{Q}, \mathcal{D}, \mathcal{R} \rangle$ specifying the required information for the assignments, and X, the current solution.
Output: X' a neighbour of X.

1: Randomly select a skill k_a in the solution X.
2: For skill k_a, delete all the assignments connecting people with the projects in X.
3: Apply Algorithm 1 for finding a new feasible assignment X' between people in k_a and the projects.

Note also that if X' is a neighbour of X then the Global Efficiency of X', i.e., $E(X')$, must be evaluated again since X' (and also any X) involve interactions with people from other skills.

The neighbourhood N^1 can be generalised to N^k according to the number $1 \le k \le f$ of skills chosen at the beginning and whose people are reallocated. For solving practical problems in the context of the MTFP, the number of skills should be a relatively small number (e.g., $k < 10$). For example N^3 means that three skills are selected, at random, and their people are reallocated. The Local Search using N^1 is denoted by $\mathrm{LS}(N^1, X)$, whereas the Variable Neighbourhood Search presented later will be denoted as $\mathrm{VNS}(N^k, X)$.

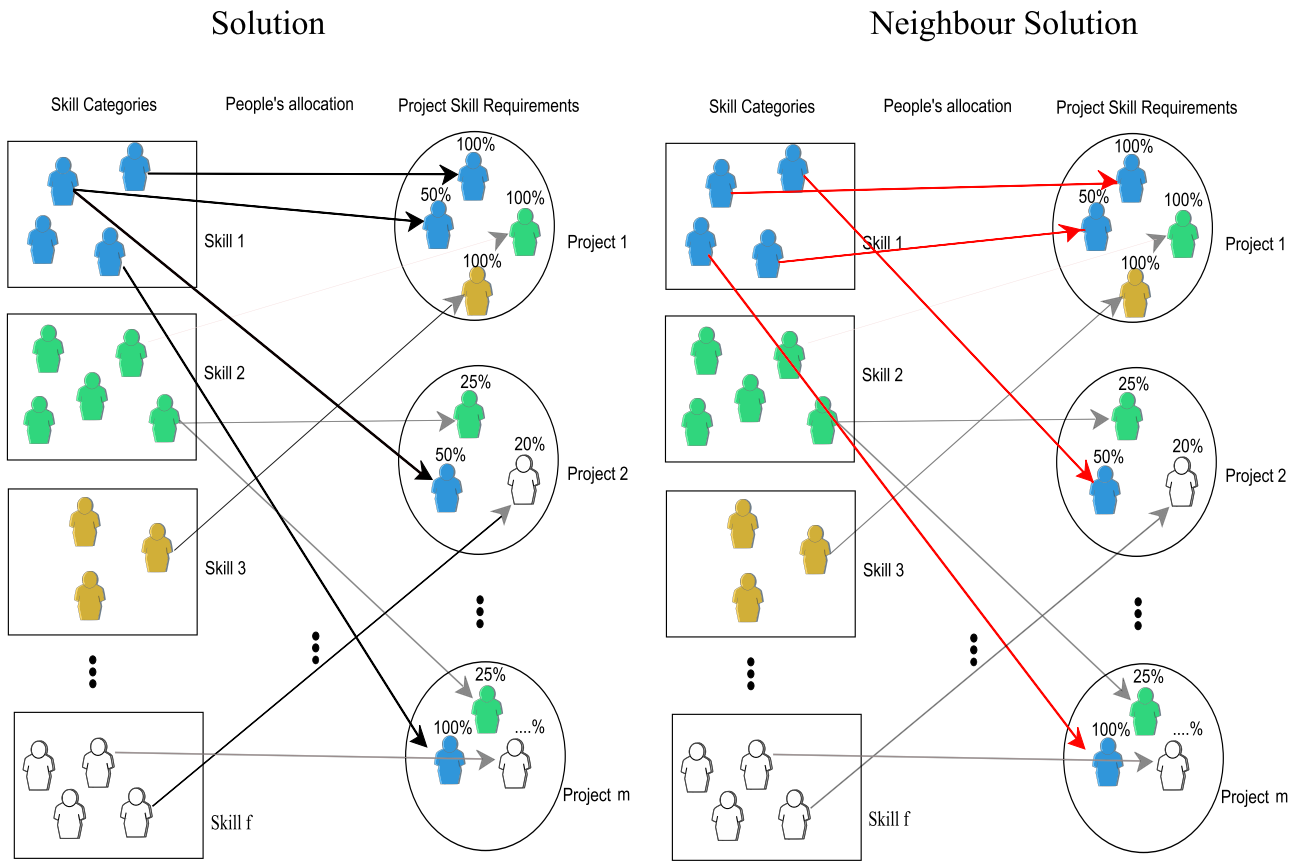


Fig. 2. Illustration of the neighbourhood N^1 using skill 1. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

Algorithm 3. LS(N^1, X).

Input: M=Maximum number of iterations, $\mathcal{Q}$, $\mathcal{D}$, $\mathcal{R}$, $\mathcal{S}$, $\mathcal{W}$.
Output: A local optimum X and $E(X)$.

$X_0 \leftarrow$ *Initial solution*
$E(X_0) \leftarrow$ Efficiency of X_0
i:=0
X:=X_0
while $i < M$ **do**
  $X' \leftarrow N^1(X)$
  if $E(X') > E(X)$ **then**
    X:=X'; $E(X)$:=$E(X')$
  i:=$i + 1$

4.3. Variable Neighbourhood Search Metaheuristic (VNS) for the MTFP

The VNS metaheuristic is illustrated in Algorithm 4. At first, VNS performs a local search using neighbourhood N^1 until reaching a local optimum X. From this point on, the algorithm sequentially explores other higher k neighbourhoods going back (eventually) to the first k=1 neighbourhood once a neighbourhood N^k with $k > 1$ skills finds an allocation X' such that $E(X') > E(X)$. In an extreme case all the f neighbourhoods will be used, where f is the total number of skills.

The worst case time complexity is $O(M'fM)$, where M' is a parameter of the VNS, f is the total number of skills, and M is a parameter of the LS algorithm.

Algorithm 4. Variable neighbourhood search k-skill.

Input: M' = Maximum number of iterations, $\mathcal{Q}$, $\mathcal{D}$, $\mathcal{R}$, $\mathcal{S}$, $\mathcal{W}$.
Output: A local optimum X and $E(X)$.

$X_0 \leftarrow$ *Initial solution*
$E(X_0) \leftarrow$ Efficiency of X_0
i:=0
X:=X_0
while $i < M'$ **do**
  $k \leftarrow 1$
  j:=1
  while $j \le f$ **do**
    $X' \leftarrow LS(N^k, X)$
    if $E(X') > E(X)$ **then**
      X:=X'; $E(X)$:=$E(X')$
      k:=1
    else
      $k \leftarrow k + 1$
    j:=$j + 1$
  i:=$i + 1$

5. Computational experiments and results analysis

In this section, three types of problems as a function of the percentage of positive relationships among the people available are defined and tested. The experimental performance of the three algorithms in these three groups of instances is registered and the experimental results are analysed.

Table 6
Generic particular variable values for each group type of instances.

Classes of Instances	Projects (m)	People available (n)	Skills (f)	Time allocation fractions (t)
1	2	25	10	0.0–1.0
2	5	50	5	0.0–1.0
3	10	100	10	0.0–1.0
4	2	25	10	0.0–0.5–1.0
5	5	50	5	0.0–0.5–1.0
6	10	100	10	0.0–0.5–1.0
7	2	25	10	0.0–0.25–0.50–0.75–1.0
8	5	50	5	0.0–0.25–0.50–0.75–1.0
9	10	100	10	0.0–0.25–0.50–0.75–1.0

5.1. Test problems and machine implementation

Three groups of instances were created. The criterion used for the classification was the percentage of the positive relationships in the sociometric matrix: 20% for the instances group type I, 30% for the instances group type II and 50% for the instances group type III. This criterion was used due to the high influence over the probability of finding solutions with high efficiency. Within each group (I, II and III), 9 classes of instances varying the values of other variables (number of projects, number of people available, number of skills, minimum time allocation allowed) were created. Finally, for each of the 9 classes of instances within each of the three group types with different positive relationships, 10 specific instances were randomly generated. Table 6 shows the problem structure devised for each group of instances.

All the experiments were performed on a Intel Core i5 machine with 8 GB RAM.

5.2. Algorithmic performance and analysis

Table 7 shows the average results obtained by the proposed algorithms when applied to the 27 classes of instances. For the sake of clarity, it is worth noting that the CP algorithm experiments were always run for a fixed time of 5 minutes, since early experiments detected that, without exception, CP efficiency results barely improve after that time limit. Concerning LS and VNS algorithms, they were tested by limiting the maximum number of iterations at $M=5000$, that, as shown later in Fig. 4, corresponds to a number of iterations by which both algorithms generally reach efficiency values close enough to an asymptotic value. Running this number of iterations always took LS less than 100 s (49 s on average) and VNS less than 200 s (96 on average). Results in Table 7 show that, even though VNS algorithm requires much less than 5 min to perform all the calculations, its global efficiency values are generally slightly above CP algorithm's.

In this regard, Table 8 shows the Anova results comparing the algorithm efficiency averages by couples (CP vs LS, CP vs VNS and LS vs VNS). It can be easily seen that whenever the $|T|$ statistic values are above the critical t-student values for $\alpha = 5\%$, the efficiency averages of the considered algorithms for that instance cannot be considered equal. A quick analysis reveals that, despite most differences between the CP and LS results are not significant, approximately in one out of three occasions, VNS efficiency results are significantly above CP. The latter added to the fact that only in one out of nine occasions CP outperforms VNS, concludes that the

Table 7
Average algorithmic performance for the nine instances of the three group types (values in bold face represent the best solutions).

Group type	Classes	CP		LS		VNS		Efficiency average
		Efficiency	σ	Efficiency	σ	Efficiency	σ	
	I.1	0.37	0.01	0.37	0.01	**0.39**	0.01	0.38
	I.2	0.27	0.01	0.26	0.02	**0.30**	0.01	0.28
	I.3	0.30	0.01	0.31	0.01	**0.33**	0.02	0.31
	I.4	0.40	0.03	0.38	0.01	**0.41**	0.01	0.40
I	**I.5**	0.31	0.04	**0.32**	0.01	**0.32**	0.02	0.32
	I.6	0.33	0.00	0.33	0.01	**0.34**	0.01	0.33
	I.7	**0.42**	0.03	0.40	0.01	0.40	0.01	0.41
	I.8	0.34	0.09	0.33	0.01	**0.35**	0.01	0.34
	I.9	0.33	0.01	**0.34**	0.02	**0.34**	0.02	0.34
	Average	0.34		0.34		0.35		0.34
	II.1	**0.92**	0.02	**0.92**	0.03	**0.92**	0.03	0.92
	II.2	0.80	0.01	0.77	0.15	**0.84**	0.02	0.80
	II.3	**0.67**	0.06	0.65	0.02	**0.67**	0.04	0.66
	II.4	**1.00**	0.00	0.96	0.05	**1.00**	0.00	0.99
II	**II.5**	**0.85**	0.05	0.84	0.10	0.84	0.02	0.84
	II.6	**0.79**	0.06	0.76	0.15	**0.79**	0.01	0.78
	II.7	**1.00**	0.00	**1.00**	0.05	**1.00**	0.00	1.00
	II.8	0.87	0.03	0.90	0.10	**0.92**	0.01	0.90
	II.9	**0.83**	0.04	**0.83**	0.15	**0.83**	0.01	0.83
	Average	0.86		0.85		0.87		0.86
	III.1	0.85	0.06	0.80	0.08	**0.90**	0.03	0.85
	III.2	**0.82**	0.04	**0.82**	0.10	**0.82**	0.01	0.82
	III.3	0.88	0.05	0.84	0.15	**0.90**	0.02	0.87
	III.4	0.93	0.03	0.93	0.05	**0.94**	0.02	0.93
III	**III.5**	**0.90**	0.02	**0.90**	0.04	**0.90**	0.01	0.90
	III.6	0.94	0.02	0.92	0.01	**0.97**	0.03	0.94
	III.7	**1.00**	0.00	**1.00**	0.01	**1.00**	0.00	1.00
	III.8	**0.98**	0.02	0.97	0.01	0.97	0.01	0.97
	III.9	**1.00**	0.00	**1.00**	0.00	**1.00**	0.01	1.00
	Average	0.92		0.91		0.93		0.92

Table 8
Results of the unpaired two-sample t-test for equal means assuming unequal variances ($|T| > t$ = *Yes* denotes means that cannot be considered equal with $\alpha = 5\%$, cases with "–" indicate σ=zero).

Classes	**CP vs LS**				**CP vs VNS**				**LS vs VNS**			
	$\|T\|$	$t_{1-\alpha/2,df}$	df	$\|T\| > t$?	$\|T\|$	$t_{1-\alpha/2,df}$	df	$\|T\| > t$?	$\|T\|$	SS	MS	$\|T\| > t$?
I.1	0.00	2.10	18	No	4.47	2.10	18	Yes	4.47	2.10	18	Yes
I.2	1.41	2.16	13	No	6.71	2.10	18	Yes	5.66	2.16	13	Yes
I.3	2.24	2.10	18	Yes	4.24	2.16	13	Yes	2.83	2.16	13	Yes
I.4	2.00	2.23	11	No	1.00	2.23	11	No	6.71	2.10	18	Yes
I.5	0.77	2.23	10	No	0.71	2.16	13	No	0.00	2.16	13	No
I.6	0.00	2.26	9	No	3.16	2.26	9	Yes	2.24	2.10	18	Yes
I.7	2.00	2.23	11	No	2.00	2.23	11	No	0.00	2.10	18	No
I.8	0.35	2.26	9	No	0.35	2.26	9	No	4.47	2.10	18	Yes
I.9	1.41	2.16	13	No	1.41	2.16	13	No	0.00	2.10	18	No
II.1	0.00	2.13	16	No	0.00	2.13	16	No	0.00	2.10	18	No
II.2	0.63	2.26	9	No	5.66	2.16	13	Yes	1.46	2.26	9	No
II.3	1.00	2.23	11	No	0.00	2.13	16	No	1.41	2.16	13	No
II.4	2.53	2.26	9	Yes	–	–	–	No	2.53	2.26	9	Yes
II.5	0.28	2.16	13	No	0.59	2.20	12	No	0.00	2.26	10	No
II.6	0.59	2.20	12	No	0.00	2.26	9	No	0.63	2.26	9	No
II.7	0.00	2.26	9	No	–	–	–	No	0.00	2.26	9	No
II.8	0.91	2.23	11	No	5.00	2.23	11	Yes	0.63	2.26	9	No
II.9	0.00	2.23	10	No	0.00	2.23	10	No	0.00	2.26	9	No
III.1	1.58	2.12	17	No	2.36	2.16	13	Yes	3.70	2.20	11	Yes
III.2	0.00	2.20	12	No	0.00	2.23	10	No	0.00	2.26	9	No
III.3	0.80	2.23	11	No	1.17	2.20	12	No	1.25	2.26	9	No
III.4	0.00	2.14	15	No	0.88	2.13	16	No	0.59	2.20	12	No
III.5	0.00	2.16	13	No	0.00	2.16	13	No	0.00	2.23	10	No
III.6	3.51	2.13	16	Yes	2.83	2.16	13	Yes	5.00	2.23	11	Yes
III.7	0.00	2.26	9	No	–	–	–	No	0.00	2.26	9	No
III.8	1.41	2.16	13	No	1.41	2.16	13	No	0.00	2.10	18	No
III.9	–	–	–	No	0.00	2.26	9	Yes	0.00	2.26	9	Yes

VNS algorithm slightly outperforms VNS most of the time.

Concerning problem complexity, it is clear from Table 7 that instances from group type III achieve Efficiency values over 80% in all cases. On the other hand, instances from group type II evidence higher dispersion varying between 67% and 100%, whereas instances of group type I display efficiency values that rarely achieve 40%. In this regard, the absence of optimum solutions found is probably the consequence of their extreme scarcity due to the insufficient number of positive relationships among the allocated members. Therefore, it is possible to state that it is very difficult to find solutions with high efficiency when the proportion of positive relations is too low, an intuitive outcome that was to be expected beforehand. Analogously, despite not being proved here, groups of instances with percentage above 50% would have produced almost steadily maximum efficiencies.

Also, it is easy to note that each class of instances within each group type evidence important differences at times in the levels of efficiency, clearly denoting that different values of other variables such as number of projects, number of people available, number of skills and minimum time allocation allowed, are influential as well, but not to the same extent. This particular issue will be addressed later by Fig. 4.

Finally, Fig. 3 only illustrates the performance of the VNS algorithm, as being the one that outperformed the other two algorithms. However, the graphs corresponding to CP and LS keep a similar distribution.

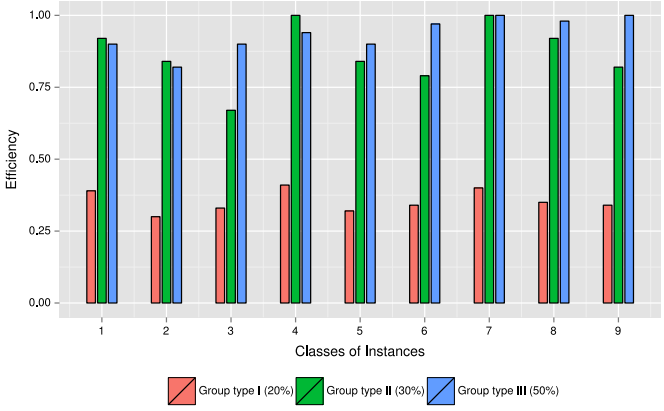
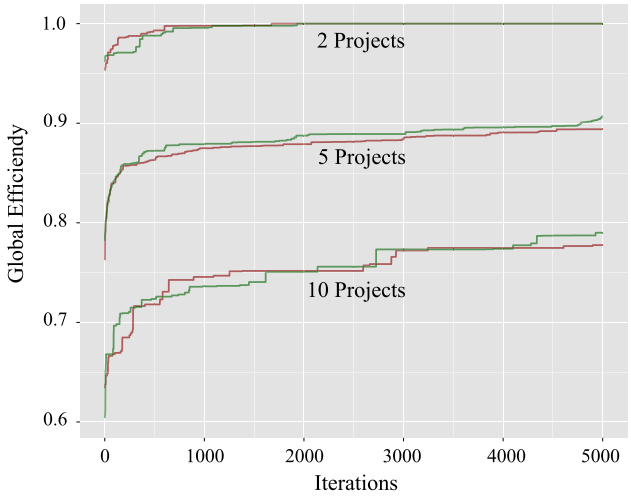


Fig. 3. Efficiency of VNS for the three different types and classes of instances.

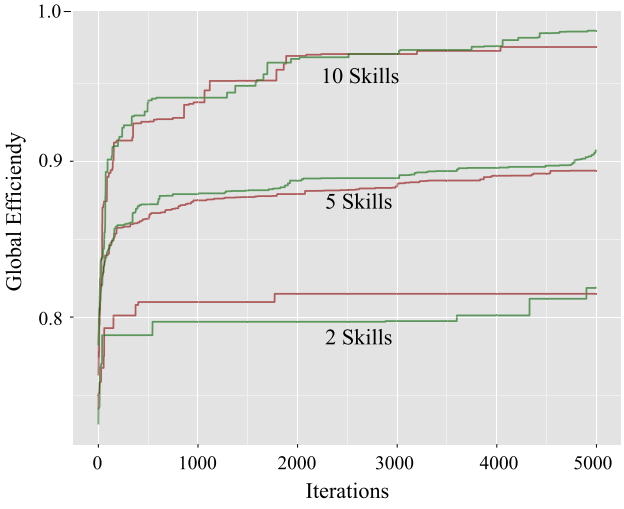
5.2.1. *Sensitivity analysis*

Fig. 4 shows the performance of LS and VNS algorithms over 5000 iterations – CP was not included since its performance evolution cannot be extracted from the commercial software used. The five graphs are associated each to the variation of one problem parameter value at a time with respect to the baseline instance tagged as group type II and class 5. This particular baseline instance, which is the same that was used earlier in Section 3.2 to illustrate the MTFP notation, considers 50 people available categorized into 5 skills which are allocated to 5 projects. The importance (weight) of each project was directly calculated according to the number of people each project required, and the sociometric matrix was artificially created in such a way to satisfy the requirement that at least one set of people pre-assigned to each project always had positive relationships $(+1)$ with each other, while filling the remaining sociometric matrix elements randomly but taking into account that the total percentage of $+1$ relationships should add up 30%. Hence, at least one optimum solution exists with the maximum possible Global Efficiency (E=1.00), but, of course, the algorithms do not know what it is.

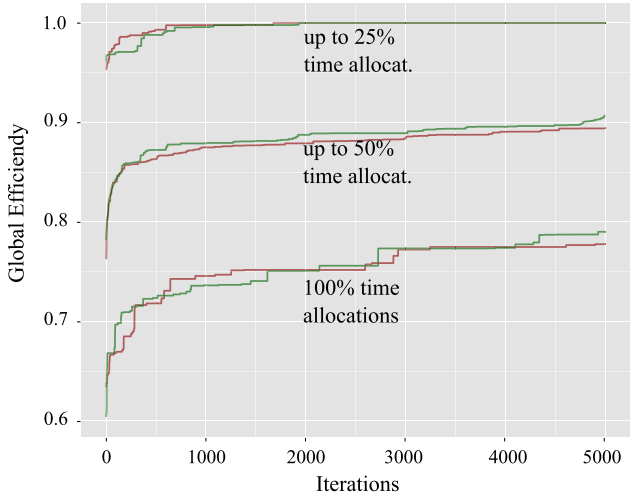
Therefore, a sensitivity analysis was applied to a basic instance by making changes in one variable, both increasing and decreasing its value. In Fig. 4a changes on the number of projects was done by considering 2, 5 and 10 projects. In Fig. 4b the changes were applied to the number of skills by considering 2, 5 and 10 skills. Fig. 4.c illustrates the results by changes in the minimum people's



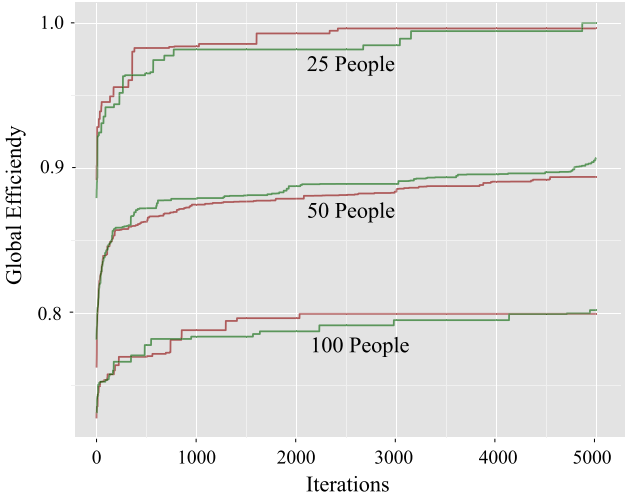
(a) Different number of projects.



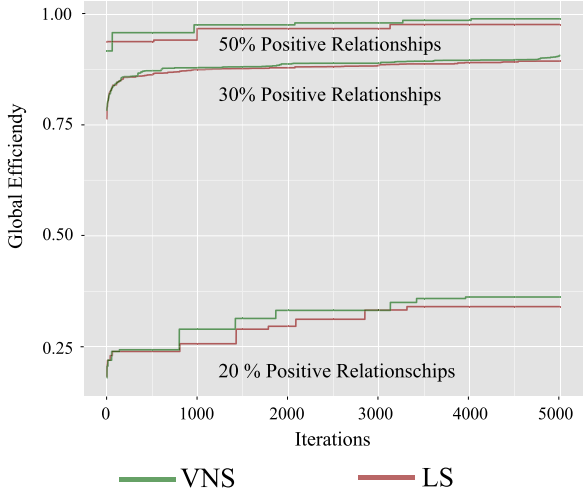
(b) Different number of skills.



(c) Different time allocations.



(d) Different number of people.



(e) Different number of positive relationships.

Fig. 4. Sensitivity analysis for the LS (shown in red) and VNS (shown in green) algorithms applied to the MTFP. Each plot shows three examples using different values for the corresponding parameter: (a) Different number of projects. (b) Different number of skills. (c) Different time allocations. (d) Different number of people. (e) Different number of positive relationships. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

time allocations fractions of 25%, 50% and 100%. Fig. 4d considers changes on the number of people, 25, 50 and 100, and Fig. 4e considers changes on the percentage of positive relationships, 20%, 30% and 50%.

Therefore, the algorithms must compete, and the closer they approach 1, the better their performance is. In this case, using the same problem formulation stated above, the algorithms were run ten times each as noted earlier, and the resulting average evolution of Global Efficiency is depicted in Fig. 4 as a function of the number of iterations.

A first reading shows that as the number of people and number of projects increase, as well as the percentage of positive relationships decreases, the problem becomes more difficult. However, the other two graphs show results that appear to be counterintuitive: when the number of skills decreases and the minimum allocation time approaches 1.00, the problem also becomes more difficult. The cases for larger numbers of people, projects and positive relationships are straightforward; nevertheless, the latter cases require further explanation.

Particularly, when the number of skills decreases, there is a larger amount of people within the same skill, which means that the number of permutations of the integrated elements (people) to be chosen within the same skill is greater. The extreme would be a case with one skill in which all people had to be chosen from that same group (the case with the highest number of possible combinations). The opposite case would be one with a skill categorisation for which all skills encompassed a single person each. In this case, the solution would be nearly unique if projects demanded people from each skill and is therefore trivial.

However, the observation that situations in which people can only be allocated full-time lead to lower Global Efficiencies is quite surprising because it is obvious that the number of combinations for the cases in which people can be fragmented over more projects have far more possible combinations, and therefore, it should be harder when indeed it is not. The authors consider that this unexpected outcome has to be necessarily related to a likely lower quantity of good solutions located near the optimum/s as a consequence of the more strict (radical) people allocations (either a person is allocated to one project or another, but not partially to some simultaneous projects, which could give hints concerning where a better solution can be found).

Finally, for comparison of the algorithms themselves, the results verify that the Variable Neighbourhood metaheuristic always performs better than the other algorithms with the only exception of the 100-people problem in which VNS is ranked in a close second position. Therefore, Variable Neighbourhood algorithms appear to be a good alternative for solving this kind of problem, even though their superiority is not significant and performance issues occur as the problem grows in size, thus requiring further improvement.

6. Conclusions and future work

The Multiple Team Formation Problem (MTFP) can be understood as the problem of allocating multiple people (either full-time or in smaller time fractions) categorized into one or several skills to multiple teams or projects (groups) that require a specified amount of people per skill. This problem provides interesting analogies with real-life situations, e.g., environments in which project managers must allocate several workers on-hand to multiple simultaneous projects (each requiring multidisciplinary skills) while at the same time attempting to maximise the positive social relationships among those workers allocated to the same project, because it is widely accepted that group cohesion boosts motivation, and higher levels of motivation lead to higher productivity and performance standards.

Perhaps due to the combination of both the importance of this problem and the recent development of social network analysis, the TFP has become an Operational Research problem of high interest since 2009, when the first attempt at solving this problem for a single project or team was tackled. During the last years, however, the number of publications related to this problem has steadily increased. Nevertheless, this is the first paper to thoroughly address the problem of allocating part-time people to multiple teams, named here as Multiple Team Formation Problem (MTFP) and to propose a first OR mathematical programming formulation.

Besides, three algorithms have been implemented to address the problem of finding a good solution to the MTFP, being the Variable Neighbourhood Search algorithm the best approach found thus far, even though it only slightly outperforms the other two algorithms implemented: Constraint Programming and Local Search. Furthermore, despite the three implemented algorithms perform well in middle-sized problems, it comes clear that there is still room for improvement for problems with higher values of number of people and projects involved, as well as for problems with fewer number of positive interactions among the people to be allocated.

Finally, it is worth noting that multiple research paths are now available in addition to the need for devising better algorithms to solve this problem. First, research on how to fill out the sociometric matrix for the first time and keep it updated would be interesting as an alternative to forcing the people available to undergo social queries about their own preferred and non-preferred partners. Second, an interesting extension would consist of allowing the people involved to be categorized under more than one skill, an issue that has already been considered for single-team formation but certainly not for multiple teams since it avoids the problem decomposition into feasible solutions per skill. Third, another interesting line of research would be to develop algorithms and/or set boundary conditions that allow a minimum Global Efficiency value to be exceeded for all projects/teams created regardless of whether the overall Global Efficiency was optimal.

All of these interesting continuity options make clear that the Multiple Team Formation approach proposed in this paper has only begun to reveal the depths of its complexity.

Acknowledgements

This research study was funded in Chile by CONICYT under the Programs Initiation Into Research 2012, 2013 and 2014 (Project numbers 11121350, 11130666 and 11140128).

Open Access of this manuscript was funded by the School of Construction Management and Engineering of the University of Reading (United Kingdom).

References

[1] Agrawal R, Golshan B, Terzi E. Grouping students in educational settings. In: Proceedings of the 20th ACM SIGKDD international conference on knowledge discovery and data mining – KDD '14. New York, USA: ACM Press; 2014. p. 1017–26.

[2] Agustín-Blas LE, Salcedo-Sanz S, Ortiz-García EG, Portilla-Figueras A, Pérez-Bellido AM, Jiménez-Fernández S. Team formation based on group technology: a hybrid grouping genetic algorithm approach. Comput Oper Res 2011;38 (2):484–95.

[3] Aringhieri R. Composing medical crews with equity and efficiency. Cent Eur J Oper Res 2009;17(3):343–57.

[4] Aringhieri R, Landa P, Soriano P, Tànfani E, Testi A. A two level metaheuristic for the operating room scheduling and assignment problem. Comput Oper Res

2015;54:21–34.
[5] Awal GK, Bharadwaj KK. Team formation in social networks based on collective intelligence – an evolutionary approach. Appl Intell 2014;41(2):627–48.
[6] Babad E. On the conception and measurement of popularity: more facts and some straight conclusions. Soc Psychol Educ 2001;5(1):3–29.
[7] Bailey DE. Manufacturing improvement team programs in the semiconductor industry. Semicond Manuf, IEEE Trans Semicond Manuf Addresses Chall Probl Manuf Complex Microelectron Components 1997;10(1):1–10.
[8] Ballesteros-Pérez P, González-Cruz MC, Fernández-Diego M. Human resource allocation management in multiple projects using sociometric techniques. Int J Proj Manag 2012;30(8):901–13.
[9] Ballesteros-Pérez P, González-Cruz MC, Pastor-Ferrando J. Analysis of construction projects by means of value curves. Int J Proj Manag 2010;28(7):719–31.
[10] Ballesteros-Pérez P, González-Cruz MdC, Fernández-Diego M. Proposal for a new method using sociometric techniques to form optimal work groups. In: Proceedings of the 18th international congress on project management and engineering; 2014. p. 01–002.
[11] Ballesteros-Pérez P, González-Cruz M, Fernández-Diego M, Pellicer E. Estimating future bidding performance of competitor bidders in capped tenders. J Civil Eng Manag 2014;20(5):702–13.
[12] Baykasoglu A, Dereli T, Das S. Project team selection using fuzzy optimization approach. Cybern Syst 2007;38(2):155–85.
[13] Blum C, Puchinger J, Raidl GR, Roli A. Hybrid metaheuristics in combinatorial optimization: a survey. Appl Soft Comput 2011;11(6):4135–51.
[14] Campion MA, Medsker GJ, Higgs AC. Relations between work group characteristics and effectiveness: implications for designing effective work groups. Pers Psychol 1993;46(4):823–47.
[15] Caprara A, Fischetti M, Toth P, Vigo D, Guida PL. Algorithms for railway crew management. Math Program 1997;79(1–3):125–41.
[16] Cheatham M, Cleereman K. Application of social network analysis to collaborative team formation. In: Proceedings of CTS 2006 international symposium on collaborative technologies and systems; 2006. p. 306–11.
[17] Chen R-C. Grouping optimization based on social relationships. Math Probl Eng 2012 2012.
[18] Chen RC, Li JY, Ma NJ, Chang YT. Application of sociometry and genetic algorithm to selection of class officers. Inf (Jpn) 2013;16(2 A):1233–41.
[19] Croes GA. A method for solving traveling-salesman problems. Oper Res 1958;6 (6):791–812.
[20] De Backer B, Furnon V, Shaw P, Kilby P, Prosser P. Solving vehicle routing problems using constraint programming and metaheuristics. J Heuristics 2000;6(4):501–23.
[21] Delgado-Osuna J, Lozano M, García-Martínez C. An alternative artificial bee colony algorithm with destructive–constructive neighbourhood operator for the problem of composing medical crews. Inf Sci 2016;326:215–26.
[22] Dorn C, Dustdar S. On the move to meaningful internet systems: In: OTM 2010. vol. 6426 of lecture notes in computer science. Berlin Heidelberg: Springer Berlin Heidelberg; 2010.
[23] Falkenauer E. Genetic algorithms and grouping problems. John Wiley & Sons, Inc.; 1998.
[24] Farhadi F, Sorkhi M, Hashemi S, Hamzeh A. An effective expert team formation in social networks based on skill grading. In: Proceedings of 2011 IEEE 11th international conference on data mining workshops. IEEE; December 2011. p. 366–72.
[25] Farhadi F, Sorkhi M, Hashemi S, Hamzeh A. An effective framework for fast expert mining in collaboration networks: a group-oriented and cost-based method. J Comput Sci Technol 2012;27(3):577–90.
[26] Fitzpatrick EL, Askin RG. Forming effective worker teams with multi-functional skill requirements. Comput Ind Eng 2005;48(3):593–608.
[27] Gajewar A, Sarma AD. Multi-skill collaborative teams based on densest subgraphs. In: Proceedings of the 12th SIAM international conference on data mining, SDM 2012; 2012. p. 165–76.
[28] Gaston ME. Simmons J, DesJardins M. Adapting network structure for efficient team formation. In: Proceedings of AAMAS-04 workshop on learning and evolution in agent based systems; 2004. pp. 1–6.
[29] Hadj-Hamou K, Caillaud E. Cooperative design: a framework for competency-based approach. In: Proceedings of the 5th international conference on IDMME; 2004.
[30] Hansen P, Mladenović N, Pérez JAM. Variable neighbourhood search: methods and applications. Ann Oper Res 2010;175(1):367–407.
[31] Hendriks M, Voeten B, Kroep L. Human resource allocation in a multi-project R&D environment. Int J Proj Manag 1999;17(3):181–8.
[32] Hollenbeck JR, DeRue DS, Guzzo R. Bridging the gap between I/O research and HR practice: improving team composition, team training, and team task design. Hum Resour Manag 2004;43(4):353–66.
[33] Hooker JN. A principled approach to mixed integer/linear problem formulation. In: Operations research and cyber-infrastructure. Springer; 2009. p. 79–100.
[34] Jacob LM, Helen HJ. The sociometry reader.Glencoe (Illinois): The Free Press; 1960. p. 35.
[35] Johnson DS, Aragon CR, McGeoch LA, Schevon C. Optimization by simulated annealing: an experimental evaluation; part II, graph coloring and number partitioning. Oper Res 1991;39(3):378–406.
[36] Kirkpatrick S. Optimization by simulated annealing: quantitative studies. J Stat Phys 1984;34(5–6):975–86.
[37] Lappas T, Liu K, Terzi E. Finding a team of experts in social networks. In: Proceedings of the 15th ACM SIGKDD international conference on knowledge discovery and data mining – KDD '09. New York, USA: ACM Press; 2009. p. 467.
[38] Li C-T, Shan M-K. Team formation for generalized tasks in expertise social networks. In: Proceedings of 2010 IEEE second international conference on social computing. IEEE; August 2010. p. 9–16.
[39] Li C-T, Shan M-K, Lin S-D. On team formation with expertise query in collaborative social networks. Knowl Inf Syst 2013;42(2):441–63.
[40] Lin C, Gen M. Multi-criteria human resource allocation for solving multistage combinatorial optimization problems using multiobjective hybrid genetic algorithm. Expert Syst Appl 2008;34(4):2480–90.
[41] Liu C, Yang S. A hybrid genetic algorithm for integrated project task and multi-skilled workforce scheduling. J Comput Inf Syst 2011;7(6):2187–94.
[42] Maurer I. How to build trust in inter-organizational projects: the impact of project staffing and project rewards on the formation of trust, knowledge acquisition and product innovation. Int J Proj Manag 2010;28(7):629–37.
[43] Mladenović N, Hansen P. Variable neighborhood search. Comput Oper Res 1997;24(11):1097–100.
[44] Mora-Melia D, Iglesias-Rey P, Martinez-Solano F, Ballesteros-Pérez P. Efficiency of evolutionary algorithms in water network pipe sizing. Water Resour Manag 2015;29(13):4817–31.
[45] Moreno JL. Foundations of sociometry: an introduction. Sociometry 1941;4 (1):15–35.
[46] Pesant G. A constraint programming primer. EURO J Comput Optim 2014; (2):89–97.
[47] Russell SJ, Norvig P. Artificial Intelligence: a modern approach. 3rd edPearson Education; 2009.
[48] Shi Z, Hao F. A strategy of multi-criteria decision-making task ranking in social-networks. J Supercomput 2013;66(1):556–71.
[49] Sorkhi M, Alvari H, Hashemi S, Hamzeh A. A game-theoretic framework to identify top-k teams in social networks. In: KDIR 2012 – proceedings of the international conference on knowledge discovery and information retrieval; 2012. p. 252–7.
[50] Teixeira LO, Huzita EHM. Distributed computing and artificial intelligence. in: Proceedings of 11th international conference, vol. 290 of advances in intelligent systems and computing. Springer International Publishing, Cham. 2014.
[51] Van Laarhoven PJ, Aarts EH, Lenstra JK. Job shop scheduling by simulated annealing. Oper Res 1992;40(1):113–25.
[52] Wi H, Oh S, Mun J, Jung M. A team formation model based on knowledge and collaboration. Expert Syst Appl 2009;36(5):9121–34.
[53] Xing B, Gao W-J. Innovative computational intelligence: a rough guide to 134 clever algorithms. Springer Publishing Company; 2014.
[54] Yin H, Cui B, Huang Y. Advanced data mining and applications. Vol. 7120 of lecture notes in computer science. Berlin Heidelberg: Springer Berlin Heidelberg; 2011.
[55] Zzkarian A, Kusiak A. Forming teams: an analytical approach. IIE Trans 1999;31(1):85–97.