J Sched (2012) 15:363–380
DOI 10.1007/s10951-011-0245-x

An improved MIP-based approach for a multi-skill workforce scheduling problem

Murat Fırat · C.A.J. Hurkens

Published online: 4 August 2011

Abstract This paper deals with scheduling complex tasks with an inhomogeneous set of resources. The problem is to assign technicians to tasks with multi-level skill requirements. Here, the requirements are merely the presence of a set of technicians that possess the necessary capabilities. An additional complication is that a set of combined technicians stays together for the duration of a work day. This typically applies to scheduling of maintenance and installation operations. We build schedules by repeated application of a flexible matching model that selects tasks to be processed and forms groups of technicians assigned to combinations of tasks. The underlying mixed integer programming (MIP) model is capable of revising technician-task allocations and performs very well, especially in the case of rare skills.

Keywords Project scheduling · Multi-skill workforce scheduling · Mixed integer programming

1 Introduction

As specialization in production and maintenance increases, the importance of skill management in employee scheduling grows significantly. Especially when activities require skills from several specialization fields at different levels, skill management becomes more challenging. Multi-skilled employee scheduling can be encountered, for example in companies having operations like maintenance, construction and installation in which the work is carried out at different physical locations. Then it makes sense to keep a combined group of workers together for a workday.

The scheduling problem under consideration is the 2007 ROADEF Challenge problem. It falls in the class of "resource-constrained project scheduling problems" (RCPSP). It has some additional aspects increasing the complexity and making it impossible to use other approaches for RCPSP in the literature. We developed an approach to this problem based on a hybrid combination of MIP models and applied it on maintenance instances provided by France Telecom in the 2007 ROADEF Challenge. Since the problem instances of France Telecom have been used as a test bed for a computational challenge held in 2007, we can compare the performance of our method to other approaches tackling the same problem.

In the problem, we are given a set of tasks and a group of technicians. In each workday, groups of technicians or teams are supposed to perform workloads that are merely task sequences. A team formed on a certain workday to carry out a task must stay together for the duration of that workday. In a teamload, tasks must be performed without overlapping (one at a time), without interruption, and must start and end on the same day. Any travel or setup time in between tasks is disregarded.

Among tasks, there may be precedence relations requiring that a certain task, say p, must be completed before another task, say q, can be processed. In this case we say p precedes q or q succeeds p. A task can be processed by a group of technicians provided that the collective capabilities of this group are above a certain threshold. The capabilities or skills required by tasks are described in terms of *domains* and *levels*. Moreover, a certain budget may be provided for outsourcing some tasks to finish the project in shorter time.

Supported by France Telecom/TUE Research agreement No. 46145963.

M. Fırat (✉) · C.A.J. Hurkens
Department of Mathematics and Computer Science, TU Eindhoven, P.O. Box 513, 5600 MB Eindhoven, The Netherlands
e-mail: m.firat@tue.nl

The provided outsourcing budget should be used as much as possible, since the objective of our problem includes only weighted sum of completion times of tasks. In the problem, we are given no outsourcing information of tasks like transportation times and lead times, but only the outsourcing costs. To avoid any ambiguity, the convention of "outsourcing the successors of outsourced tasks" is used. Obviously, completion times of outsourced tasks are not taken into account.

The unavailability information of technicians is specified in the problem. In some instances, schedules may be sensitive to the absence of experts and good quality solutions can be obtained by deciding to outsource the right combination of tasks and/or by efficient packing of the expert-requiring tasks on the days when experts are available.

Tasks are partitioned into several priority classes depending on their urgencies. The latest completion time of tasks under a priority class is called the "priority span." A priority class contributes to the schedule cost with its weight multiplied by its priority span. The total contribution of priority classes determines the cost of a schedule and the objective of our scheduling problem is to minimize the cost.

The proposed combinatorial approach is composed of two phases: *preprocessing* and *schedule construction*. The preprocessing phase has two parts. In the first part, several key properties of tasks are calculated. In the second part, lower bounds are computed and outsourced tasks are determined. We solve a simplified problem in which skill requirements and precedence relations of tasks are relaxed, preemption is allowed, unavailability of technicians is taken into account, and the option of outsourcing tasks is preserved. The problem is formulated as a MIP model which assumes that priority classes must be completed in a predetermined order. This order is given by the specified permutation of priority classes (for brevity: "priority permutation").

In the schedule construction phase, a number of alternative schedules are built by varying several parameters and strategies. One important parameter for a schedule is the priority permutation, since priority classes are handled sequentially while constructing a schedule. It is assumed that priority permutations with smaller lower bound values promise low-cost schedules. Therefore, firstly, the priority permutation with smallest lower bound value is used.

From our point of view, a schedule is composed of workday schedules due to the facts (1) tasks cannot be interrupted to perform on another workday (2) teams of technicians are formed daily. Therefore, our algorithm finds a packing of tasks (workloads of teams) greedily in the form of sequences each of not longer than a workday. The task sequences are initialized by single tasks and their lengths are increased by adding more tasks iteratively. In every iteration, the algorithm simultaneously finds enough skilled technicians for every sequence. Constructing a day schedule in this way was firstly introduced by Hurkens (2009) in the 2007 ROADEF Challenge and this approach was ranked first in the final stage among 11 qualified participants.

In this study, our main contribution is introducing the flexibility in extending the task sequences. The flexibility is maintained by three aspects of our approach (1) the sequences are allowed to get merged if necessary (2) the technicians can move easily among the groups (3) ordering of tasks in a sequence is determined by considering their precedence relations. The goal of this flexibility is being able to schedule more tasks within a workday, especially the ones requiring experts. If the expertise in the technician group is rare, as is usually the case in the companies, then the schedule cost is sensitive to the utilization of the experts. Hence, it is expected, and also supported by our computational results, that the flexibility of our approach leads to better packing of hard tasks and lower schedule costs.

The extension of task sequences is carried out by finding simultaneous technician-tasks assignments that correspond to many-to-one type matchings on the constructed bipartite graph. Matchings are restricted by skill requirements, precedence relations and total durations. The problem of finding matchings is formulated as a MIP model with limited number of variables.

The paper is organized as follows. In Sect. 2, we give the problem description. Problem complexity is analyzed in Sect. 3 and a literature review is presented in Sect. 4. Our solution approach is explained in Sect. 5. The computational results of our algorithm are comparatively reported in Sect. 6 and finally we discuss the applicability of our solution methodology to other multi-skill workforce scheduling problems in Sect. 7.

2 Problem description and notation

The problem we consider in this study was described by Dutot et al. (2006) as the contest problem in the 2007 ROADEF Challenge. In the following sections we describe the problem in our notation.

2.1 Skills

The tasks in our scheduling problem require specializations in several fields. We use the term *skill domain* for a specialization field and *skill level* to interpret the degree of expertise *hierarchically*. The set of skill domains (levels) is denoted by $\mathbb{S}$ ($\mathbb{L}$). A skill at level $l \in \mathbb{L}$ in domain $s \in \mathbb{S}$ is denoted by $(l, s) \in \mathbb{L} \times \mathbb{S}$. The skills of technicians and skill requirements of tasks are specified by matrices in $\mathbb{Z}^{\mathbb{L}\times\mathbb{S}}$.

2.1.1 *Expertise in the literature of multi-skill workforce project scheduling*

To the best of our knowledge, only Bellenguez and Neron (2004) consider hierarchical skill levels other than the participants of the 2007 ROADEF Challenge. The majority of studies treat human resources as skilled or unskilled in domains (for example Cai and Li 2000, Bellenguez and Neron 2007, Li and Womer 2009, Valls et al. 2009 and Avramidis et al. 2010). Gutjahr et al. (2008), Yoshimura et al. (2006), and Heimerl and Kolisch (2010) use competence score as an interpretation of skill level. The drawback of scoring employee expertise rather than leveling is the difficulty in expressing skill requirements of the tasks clearly. If a task requires expertise, then its skill requirement is merely a high value in a corresponding domain. This high demand may be satisfied either by assigning an expert or by collecting many non-expert employees. In case of skill leveling, the second case is not an option. Gutjahr et al. (2008) limit the number of employees that can be assigned to a task and define special variables for experts to handle this issue in their nonlinear MIP model. Yoshimura et al. (2006) use a specific parameter for experts or so-called "project leaders" in order to select an expert for each project.

Use of skills: We assume that technicians contribute simultaneously in all possible domains while processing a task and this is called *simultaneous skill use*. In the literature of multi-skill workforce project scheduling, Valls et al. (2009), Heimerl and Kolisch (2010), Drezet and Billaut (2008) and Ballou and Tayi (1996) also make the assumption of *simultaneous skill use*.

2.2 Technicians

We are given a set $T = \{1, \ldots, m\}$ of technicians to perform the tasks. The unavailability periods of technicians are considered within a finite scheduling horizon. $A(t, h)$ denotes the unavailability of technician t. It is equal to 1 if t is available on day h and zero otherwise.

Skills of a technician, say $t \in T$, are expressed by a matrix $S_t \in \{0, 1\}^{\mathbb{L}\times\mathbb{S}}$. If technician t is proficient in skill (l, s), then $S_t^{(l,s)} = 1$. Clearly, if a technician is qualified in a skill, then he is also qualified at lower levels in the domain of this skill. Hence $S_t^{(l,s)} = 1 \Rightarrow S_t^{(l',s)} = 1, \forall l' \leq l$. Once we are given S_t, the proficiency of technician t in skill domain s is found by $\max\{\{0\}, \{l \in \mathbb{L} | S_t^{(l,s)} = 1\}\}$.

A skill matrix example of a technician, say t, in the problem instance with $|\mathbb{L}| = |\mathbb{S}| = 3$ may be

$$\begin{pmatrix} 1 & 0 & 1 \\ 1 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$$

We see that technician t is expert with proficiency of level 3 in domain 1. He qualifies to skill level 2 in domain 3, but he has no skill in domain 2. A compact way to express the technician skills is using skill vector $SV \in \{0, 1, \ldots, |\mathbb{L}|\}^{\mathbb{S}}$. In the above example, the skill vector of technician t is $SV_t = (3, 0, 2)$.

Skills of teams: Let $\tau \subset T$ denote a team of technicians. The skills of the team τ are found by summing up the individual skills within τ, so we have $S_\tau = \sum_{t\in\tau} S_t$.

2.3 Tasks

In our scheduling problem, a set $J = \{1, \ldots, n\}$ of tasks is given. In this section we explain the aspects of our scheduling problem related to tasks.

Skill requirements: Tasks require skill qualifications. The skill requirements of a task $j \in J$ are expressed by a matrix $RQ_j \in \mathbb{Z}^{\mathbb{L}\times\mathbb{S}}$ which provides the information of the desired skill quantity (number of technicians) and skill quality (expertise). The requirements in RQ_j are cumulative in the sense that any requirement at a level is carried to lower ones in the same domain. Therefore, for a task j and a skill (l, s) we have $RQ_j^{(l',s)} \geq RQ_j^{(l,s)}$ for all $l' \leq l$.

An example of skill requirement matrix for task j in an instance for which $|\mathbb{L}| = 3$ and $|\mathbb{S}| = 4$ may be given by

$$RQ_j = \begin{pmatrix} 1 & 2 & 0 & 0 \\ 1 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix}$$

According to the given skill requirement example, in a team processing task j, there must be at least two technicians qualified in domain 2, one being proficient at least at level 2 and one at least at level 1. Let $T(j) \subset T$ be such a team. Consequently, the team $T(j)$ must satisfy $S_{T(j)}^{(l,s)} \geq RQ_j^{(l,s)}$ for all $(l, s) \in \mathbb{L} \times \mathbb{S}$.

Durations: The time needed to perform task j is called its duration and denoted by d_j. The duration of each task is fixed and does not vary with the number and expertise of technicians assigned. Processing of tasks cannot be interrupted, and if a team started performing a task, that team must finish the task within the same workday. This also implies that $d_j \in \{1, 2, \ldots, H\}$ where H is the workday length.

Precedence relations: Precedence relations of task j enforce that all tasks in $Pred(j)$ must be completed before the task j starts. A task $k \in Pred(j)$ is said to be a *predecessor* of task j and their relation is denoted by $k \rightarrow j$. Moreover, task j is also said to be a *successor* of task k. Let CT_j be the completion time of task j in a schedule. Precedence relations of task j enforce $CT_k \leq CT_j - d_j$, for all $k \in Pred(j)$.

Outsourcing: External companies may be hired to outsource a task by paying its outsourcing cost c_j. Outsourced

tasks are discarded and need not be scheduled. The total cost of outsourced tasks must not exceed the outsourcing budget B. In the problem definition, no related information to outsourcing of tasks is given like transportation times and lead times, and hence we have the convention of outsourcing the successors of outsourced tasks. Let $\Omega \subseteq J$ denote the set of outsourced tasks. The convention of outsourcing the successors results in the following property of Ω: $\{k \in J \mid \Omega \cap Pred(k) \neq \emptyset\} \subseteq \Omega$.

Priority classes: Tasks are partitioned into several priority classes that represent urgency levels. Let P denote the set of priority classes and $P(j) \in P$ be the priority class of task j. The latest completion time of tasks under a priority class is called the priority span. It is denoted by C_p for a priority class $p \in P$ and found by $C_p = \max\{CT_j \mid P(j) = p, j \notin \Omega\}$. The overall make span is the length of a schedule and denoted by $C_0 = \max\{CT_j \mid j \in J \setminus \Omega\}$. Note that outsourced tasks do not contribute to the schedule cost. Priority class 0 is an artificial priority class that is used to include the overall makespan in the quality evaluation of schedules and of course every task belongs to priority class 0.

2.4 Schedules

In this section we explain some more notation about schedules and we state the feasibility conditions. Finally, the objective of the scheduling problem is discussed.

Workday concept: The time axis is partitioned into intervals of length H. These successive intervals represent workdays. Within a certain time interval (workday), the technicians performing a certain task must work together during this interval and they form a team. Another important restriction is that the processing of each task must stay within a time interval.

Teams: In each workday of schedules, teams of technicians are formed to process the assigned tasks. Let τ be a team on a certain workday of a schedule. $T(\tau) \subseteq T$ denotes its technicians, $J(\tau) \subseteq J$ denotes the processed tasks by the team, and $\delta(\tau) \in \{1, 2, \ldots\}$ denotes that workday. If tasks j' and j are in $J(\tau)$, then $j' <_\tau j$ denotes that j' is processed before j. In the schedules constructed by our combinatorial algorithm, we assume that there is no *idle time* between the tasks in the workloads of teams. Therefore the completion time of task j is determined as below:

$$CT_j = \big(\delta(\tau) - 1\big)H + \sum_{j' : j' <_\tau j} d_{j'} + d_j \tag{1}$$

Note that the team information consisting of sequences of tasks and groups of technicians is enough to define a solution. A workday schedule constructed by our combinatorial algorithm together with the corresponding outsourcing decision form a well-defined solution. Therefore, we do not have decision variables for starting time of tasks in our matching models.

2.4.1 Feasibility of schedules

A schedule is feasible if and only if the following constraints are satisfied:

- *Outsourcing:*

$$\sum_{j \in \Omega} c_j \leq B \tag{2}$$

The total cost of outsourced tasks must not exceed the outsourcing budget.

$$\big\{k \in J \mid \Omega \cap Pred(k) \neq \emptyset\big\} \subseteq \Omega \tag{3}$$

The successors of the outsourced tasks are outsourced as well.

- *Task and technician assignments:*

$$\big|\big\{\tau | j \in J(\tau)\big\}\big| = 1, \quad \forall j \in J \setminus \Omega \tag{4}$$

Each non-outsourced task is processed by exactly one team.

$$\big|\big\{\tau | t \in T(\tau), \delta(\tau) = h\big\}\big| \leq A(t,h), \quad \forall h, \forall t \in T \tag{5}$$

Technicians can be in at most one team on the days they are available.

- *Completion times:*

$$CT_k \leq CT_j - d_j, \quad \forall j \in J \setminus \Omega, \forall k \in Pred(j) \tag{6}$$

A successor must not start before the completion of one of its predecessors.

$$\max\big\{\big(\delta(\tau) - 1\big)H, \max\big\{CT_{j'} | j' <_\tau j\big\}\big\} \leq CT_j - d_j, \quad \forall \tau, j \in J(\tau) \tag{7}$$

Processing of tasks in the team workload must not overlap. Note that we do not allow "*idle times*" between tasks in a team workload. Therefore, the inequality sign can be replaced by equality sign for the solutions of our combinatorial algorithm.

$$CT_j \leq \delta(\tau)H, \quad \forall \tau, \forall j \in J(\tau) \tag{8}$$

Workloads of teams must not exceed the workday length.

$$C_p = \max\big\{CT_j \mid P(j) = p, j \notin \Omega\big\} \tag{9}$$

Priority span is the latest completion time of the tasks belonging to that priority.

- *Skill requirements:*

$$RQ_j^{(l,s)} \leq S_{T(\tau)}^{(l,s)}, \quad \forall \tau, \forall j \in J(\tau), \ \forall (l,s) \in \mathbb{L} \times \mathbb{S} \tag{10}$$

The technicians in every team must be enough skilled to process the assigned tasks in the workload.

A MIP model of the problem is given by Cordeau et al. (2010). The authors report that after 24-hour run of the MIP solver, an optimal schedule of even small instances with seven technicians and 20 tasks could not be found.

2.4.2 Objective

The cost of a schedule is calculated by the weighted sum of priority spans $\sum_p w(p)C_p$ where C_p denotes the priority span as expressed in (9). In the benchmark instances of France Telecom, the weights were given as $\{1, 28, 14, 4, 0\}$ for priorities $\{0, 1, 2, 3, 4\}$, respectively. The objective of our problem is to *minimize* the schedule cost. Note that the outsourcing cost is not included in the objective, therefore, in the combinatorial algorithm, we aim to use the outsourcing budget as much as possible to decrease the total work load in the schedule.

3 Problem complexity

Let us call our scheduling problem the "multi-skill technician task scheduling problem" (MTTSP). Firstly, we give a definition of MTTSP and then we prove that it is NP-Hard.

PROBLEM: MTTSP
INSTANCE: Integers H, B, M, denoting work day length, outsourcing budget, and maximum number of days of the project, respectively.
The sets T, J, P, and $\mathbb{L} \times \mathbb{S}$ denoting technicians, tasks, priority classes and skills, respectively.
For each $t \in T$; there are skills $S_t \in \{0, 1\}^{\mathbb{L}\times\mathbb{S}}$ and availability $A(t, h) \in \{0, 1\}$, for all $1 \leq h \leq M$,
For each $j \in J$; there are skill requirements $RQ_j \in \mathbb{N}^{\mathbb{L}\times\mathbb{S}}$, duration $d_j \in \{1, \ldots, H\}$, outsourcing cost c_j, predecessors $Pred(j)$, and priority class $P(j)$,
For each $p \in P$; there is a cost $w(p)$.
QUESTION: Does there exist a schedule satisfying feasibility conditions mentioned in Sect. 2.4 with cost C or less?

Theorem 1 *MTTSP is NP-Hard.*

Proof The proof is from the Subset Sum as a special case of MTTSP; see Garey and Johnson (1979) for the NP-Hardness of Subset Sum.

PROBLEM: SUBSET SUM
INSTANCE: An integer Σ and a set $A = \{a_1, \ldots, a_n\}$ in which each element a_i has size $s(a_i)$ and $\Pi = \sum_{a_i \in A} s(a_i)$.
QUESTION: Does there exist a subset $A' \subseteq A$, such that the total size of elements in A' is equal to Σ?

Let us construct a special case of MTTSP from an instance of Subset Sum as follows:

- $H = \Pi - \Sigma$, $B = \Sigma$, $M = 1$.
- $L = \{1\}$, $S = \{1\}$, and $P = \{1\}$ with $w(1) = 1$.
- $T = \{1\}$ with $S_1 = \{1\}$, $A(1, 1) = 1$.
- for every item $a_i \in A$ we create a task $j(a_i)$ with $d_{j(a_i)} = c_{j(a_i)} = s(a_i)$, $R_{j(a_i)} = \{1\}$, and $Pred(j(a_i)) = \emptyset$.

QUESTION: Does there exist a schedule with objective value H?

Note that the minimum value of the schedule cost is attained when outsourcing budget is completely used: $\sum_{j(a_i)\in\Omega} c_{j(a_i)} = B = \Sigma$ making the schedules length as well as the schedule cost $\Pi - \Sigma = H$. So a YES answer to Subset Sum leads to a YES answer for the special case of MTTSP problem. □

4 Literature review

4.1 Resource-constrained project scheduling in general

The problem considered in this paper is a generalization of the "resource-constrained project scheduling problem" (RCPSP). Brucker et al. (1999) and Hartmann and Briskorn (2010) provide extensive reviews of the RCPSP.

"Multi-mode resource-constrained project scheduling problem" (MM-RCPSP) is a generalization of the RCPSP in which activities may require renewable, non-renewable, and doubly constrained resources (see for example De Reyck and Herroelen 1999). In the "multi-skill project scheduling problem" (MSPSP) the resources are renewable human resources or staff members. Every staff member can have several skills among the needed ones by the activities. As Bellenguez and Neron (2007), and Li and Womer (2009) mentioned, MM-RCPSP formulation can be used to describe MSPSP, however, the number of combinations becomes very large even for the moderate size of employee groups, thus making it impossible to use the exact methods proposed for the MM-RCPSP to solve the MSPSP.

In the literature of project scheduling with multi-skilled human resources, several objectives are considered. For example, Avramidis et al. (2010), and Li and Womer (2009) consider minimizing staffing cost; Bellenguez and Neron (2004), Bellenguez et al. (2005), Bellenguez (2006) consider minimizing the makespan, and Wu and Sun (2006) consider minimizing the outsourcing cost. Gutjahr et al. (2008) use a hybrid objective of maximizing economic gains and personal improvement, Heimerl and Kolisch (2010) minimize both makespan and outsourcing cost.

We encounter different solution methodologies in the literature of project scheduling with multi-skilled human resources. Heimerl and Kolisch (2010) formulate an elegant

MIP model to solve project staffing and project scheduling simultaneously. Li and Womer (2009) propose a hybrid algorithm based on MIP modeling and constraint programming. The authors argue that Bellenguez and Neron (2004) do not consider personnel capacity that seems to be a result of unavailability, expertise, and other personal attributes. In another recent study, Gutjahr et al. (2008) propose a greedy heuristic as well as a hybrid solution methodology using priority-based rules, ant colony optimization and genetic algorithm to solve the so-called "project selection, scheduling and staffing with learning problem."

The project scheduling problem considered by Bellenguez and Neron (2004) shows significant similarity to our problem. In their scheduling problem, skills are expressed in domains by hierarchical levels in the same way as our problem. Skills of workers and skill requirements of jobs are specified by the same matrices as in Sect. 2.2 and in Sect. 2.3. Outsourcing of tasks is not an option in their problem and the objective is minimizing the makespan whereas we minimize a weighted sum of priority spans. Moreover, the authors assume that technicians can only work in one skill domain while performing a task contrary to our assumption "*simultaneous skill use*." This assumption seems more reasonable in cases if tasks take short time and require skills in many domains. On the other hand, if the tasks in a project require skills in several but not many domains, assigning one person to each piece of work may lead to underutilization. Consequently both problems fall in the class of the RCPSP, but differ from each other in the mentioned points.

4.2 Other solution approaches for the MTTSP

In the 2007 ROADEF Challenge, the solution approach of Hurkens (2009) was ranked first. Cordeau et al. (2010) and Estellon et al. (2009) tied for second place. Cordeau et al. (2010) describes a MIP model for our scheduling problem and the authors mention that it cannot be solved for large instances optimally in a reasonable time. They develop a meta-heuristic method that consists of a construction heuristic and an adaptive large neighborhood search with several destroy and repair methods. The solution strategy is viewed as a standard simulated annealing algorithm with a complex neighborhood search due to the acceptance criterion of the solutions.

Estellon et al. (2009) designed a local-search scheme in which a greedy algorithm is employed to obtain a feasible solution and this solution is improved by a local-search strategy. The authors use a methodology including three key points, *search strategy*, *moves* and *evaluation of moves*. They also point out that a careful implementation increases the convergence speed of local-search heuristics and stochastic elements are useful to improve the diversification.

As mentioned in the introduction, Hurkens (2009) considers the same problem and proposes a two-phase MIP-based solution methodology. In the first phase, a MIP model computes lower bound values and determines the tasks to be outsourced. Our MIP model is based on the author's one, but it is a slightly improved version in a way that tighter lower bound values for instances with heterogeneous skill distribution are found. In the second phase, two matching models are used to find technician-task assignments having limited flexibility compared to our matching models.

5 Scheduling with flexible matching model

5.1 An overview of the combinatorial algorithm

The scheduling problem under consideration is a complex problem. According to the objective, the main goal is to determine the workloads of teams, such that the number of processed tasks on each workday is maximized. As a fact in reality, the *skill quality* (or the expertise within the technician group) is limited, therefore, constructing the workloads becomes harder due to the heterogeneity in skill distribution. In such cases, the question turns to maximizing the number as well as the *hardness* of selected tasks in the workloads, since the schedule cost is sensitive to the utilization of rare expertise when skill distribution is heterogeneous.

Hurkens (2009) introduced the idea of greedily constructing the workloads, while satisfying skill requirements. The author developed a solution methodology using *matching models* which find technician-task assignments simultaneously. In this study, we improved Hurkens' solution approach by obtaining tighter lower bound values and by adding flexibility to the matching models to have a better packing of hard tasks in schedules. In this paper, we focus more on our contribution to the matching models.

Figure 1 shows the flowchart of our algorithm. The algorithm consists of two main phases; preprocessing and schedule construction. The preprocessing phase includes calculation of necessary parameters for tasks and computation of lower bounds. Computing lower bounds is merely solving a simplified problem in which skill requirements and precedence relations are relaxed, preemption is allowed, and only technician availabilities are taken into account. The simplified problem boils down to finding minimum time needed to satisfy the cumulative man-hour demand of tasks with the option of outsourcing tasks. It is assumed that the priority classes are processed sequentially and lower bounds are computed for all priority permutations of practice, since depending on the number of tasks, any priority sequence may result in lower schedule cost than the others.

In the schedule construction phase, alternative schedules are constructed by sequentially assigning priority classes.

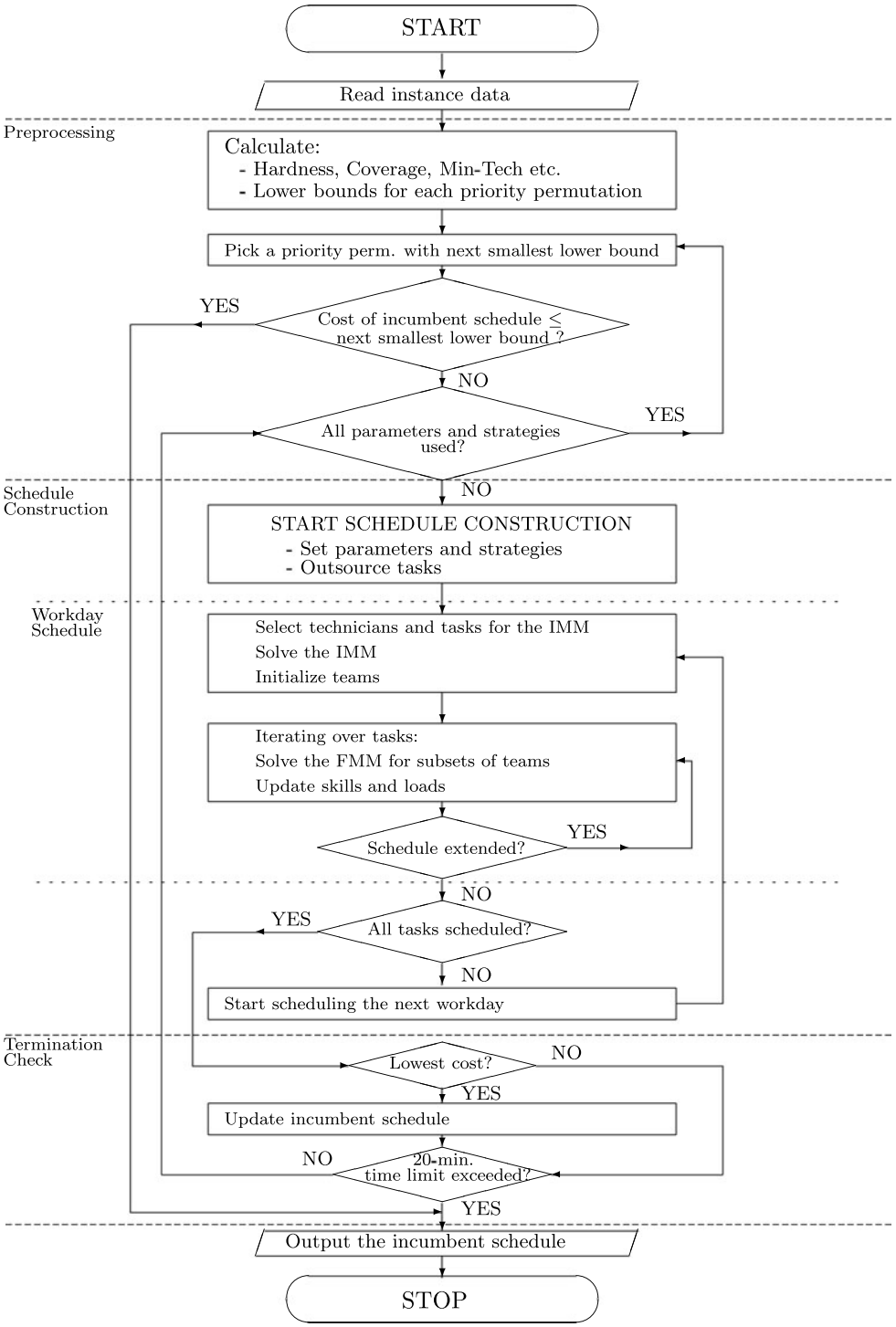

Fig. 1 Flowchart of the combinatorial algorithm

Hence, for every schedule, a certain priority permutation is fixed in advance. Priority permutations are considered according to their lower bound values: the one with smallest lower bound value is considered first, and next the second smallest and so on. Moreover, we construct more than one schedule for the same priority permutation by using several parameters and strategies for some decisions.

Having constructed a complete schedule, its cost is compared to the cost of incumbent schedule. If the cost is smaller, then the incumbent schedule is updated with the latest constructed one. Whenever the cost of the incumbent schedule is smaller than or equal to the lower bound value of the next considered priority permutation, then this priority permutation is neglected. In such a case, it is clear that

no schedule under that priority permutation can improve the schedule cost ever found. The time limit was specified as 20 minutes in the 2007 ROADEF Challenge, so the algorithm stops constructing schedules unless it is terminated before.

The combinatorial algorithm builds workday schedules successively. Constructing a workday schedule starts with the "*initial matching*". Initial matching is performed by solving a MIP model called the "initial matching model" (IMM). The model finds a many-to-one type matching on a bipartite graph in which one partition includes technicians and one partition includes tasks. Initial matching results in a partially constructed day schedule in which teams have a single task in their workloads. Next, the number of scheduled tasks in the initialized workday schedule is increased by adding more tasks greedily. We call the process of increasing the number of scheduled tasks "extending workday schedule" and this process is performed by solving a MIP model called the "flexible matching model" (FMM). Tasks can be inserted into a partial workday schedule as long as there are some teams with workload length less than a workday, skills requirements are met, and precedence relations are not violated. The algorithm starts scheduling the next workday, if no more tasks can be inserted into the current workday schedule. Constructing of workday schedules continues until all non-outsourced tasks are scheduled.

Cordeau et al. (2010) and Estellon et al. (2009) argue that the combinatorial algorithm proposed by Hurkens (2009) is an application of local search with large neighborhood exploration. However, our algorithm and Hurkens' algorithm are constructive heuristics and they construct a number of alternative schedules. Once a complete schedule is constructed, then it is not modified by any destruct and repair methods. The strategy of both algorithms is to obtain good quality solutions with the simultaneous technician-task assignments. Therefore neither algorithms can be classified as local-search algorithms.

5.2 Calculating key properties of tasks

In the preprocessing phase, several key properties of tasks are calculated by aggregating their multi-dimensional attributes. They are *min-tech*, *hardness*, *coverage*, and *matching weight*. In this section we give their formulations.

5.2.1 Min-Tech

Min-tech, denoted by MT_j, of a task j is the minimum number of technicians who can process it. This simple concept is important for two reasons: (1) man-hour demand of task j is calculated by $MH_j = MT_j d_j$ and this is used in lower bound calculations (2) in the FMM where the efficiency of an assignment is controlled by punishing the (positive) deviation from MT_j.

A simple integer programming (IP) model in (11) is used to calculate Min-Tech for each task at a time and this IP model is solved by CPLEX. The binary decision variable x_t indicates that technician t is assigned to task j. The constraints of the model enforce that skill requirements of task j are met and the objective minimizes the size of the selected team. Note that the index j is not used, since we run the following IP model for each task at a time.

$$MT_j = \min\left\{\sum_{t\in T} x_t : \sum_{t\in T} S_t^{(l,s)} x_t \geq RQ_j^{(l,s)}, \quad \forall (l,s)\in \mathbb{L}\times\mathbb{S};\ x_t \in \{0,1\}\right\} \tag{11}$$

5.2.2 Hardness

A task is said to be *hard* if it requires skills that are not common among technicians. Hence hardness of a task is a relative concept depending on the skill distribution of a technician group. For example, a task requiring moderate skill levels may be "relatively" hard for a technician group, if there are few technicians specialized in the demanded fields. Before defining the hardness, let us introduce the term "skill value," denoted by $\upsilon(l,s)$ for a skill (l,s). It is calculated as follows:

$$\upsilon(l,s) = \frac{|J_{(l,s)}|}{S_T^{(l,s)}}, \quad J_{((l,s)} = \left\{j \in J | RQ_j^{(l,s)} > 0\right\}, \quad S_T^{(l,s)} = \sum_{t\in T} S_t^{(l,s)} > 0 \tag{12}$$

Note that the value of a skill is high if the skill is rare among the technicians and also if there is some demand from tasks. Having introduced skill value, we can define "hardness," denoted by h_j for task j, by

$$h_j = \sum_{(l,s)\in\mathbb{L}\times\mathbb{S}} \upsilon(l,s) RQ_j^{(l,s)} \tag{13}$$

If the skill requirement of a task, say j, is tightly satisfied by the whole technician group, then the processing of that task is impossible when some of the necessary technicians are unavailable. It may also be very difficult to build a team for task j on a certain day, if the necessary technicians have already been allocated to several teams.

The tightness of skill satisfaction of a task can be expressed by

$$\chi_j = \max\left\{\frac{RQ_j^{(l,s)}}{S_T^{(l,s)}} \mid (l,s)\in\mathbb{L}\times\mathbb{S} : S_T^{(l,s)} \neq 0\right\} \tag{14}$$

Definition A task j with $\chi_j = 1$ is called a "special" task.

Note that $0 \leq \chi_j \leq 1$ for feasible problem instances. The ratio $RQ_j^{(l,s)}/S_T^{(l,s)}$ has a small value for common skill levels among technicians. The value of χ_j gives a sign for the relative expertise requirement and the extreme case is a special task with $\chi_j = 1$.

5.2.3 Coverage

Task j is said to *likely cover* task k if the expertise required by task k but not required by task j, is common within the technician group. Then the possibility that the team having task j in its load can also perform task k is high. Let $\alpha_{jk}^{(l,s)}$ be the pairwise comparison of skill requirements of tasks j and k for skill level (l, s). Pairwise coverage relations are defined between the tasks if the sum of their durations is less than or equal to a workday length. Note that if task j dominates the skill requirement of task k or $\alpha_{jk}^{(l,s)} \geq 0$ for all (l, s), then task j covers task k. If we have $RQ_j^{(l,s)} < RQ_k^{(l,s)}$ for some (l, s) and if these skills are common within the technician group, then it is more likely that a team performing task j can also perform task k. Pairwise comparison for each skill level between task j and k is given by

$$\alpha_{jk}^{(l,s)} := \begin{cases} 0 & \text{if } RQ_j^{(l,s)} = RQ_k^{(l,s)} = 0 \\ 1 & \text{if } RQ_j^{(l,s)} \geq RQ_k^{(l,s)} \text{ and } RQ_j^{(l,s)} \neq 0 \\ \frac{S_T^{(l,s)}}{(S_T)_{\max}} & \text{if } RQ_j^{(l,s)} < RQ_k^{(l,s)} \end{cases}$$

where $(S_T)_{\max} = \max_{(l,s)\in\mathbb{L}\times\mathbb{S}}\{S_T^{(l,s)}\}$. The coverage of task j over task k is accepted according to the expression below:

$$\gamma_{jk} := \begin{cases} 1 & \text{if } \frac{\sum_{(l,s)} \alpha_{jk}^{(l,s)}}{\sum_{(l,s)} \operatorname{sign}(\alpha_{jk}^{(l,s)})} > \varrho \\ 0 & \text{otherwise} \end{cases}$$

After some experimentation, we decided to use the tuned value $\varrho = 0.9$. Finally the coverage of task j is given by

$$\operatorname{cov}_j = \sum_{k\neq j, d_j + d_k \leq H} \gamma_{jk}$$

5.2.4 Matching weight

For the definition of matching weights to be used in the IMM and in the FMM, we consider the domination properties of tasks within the workload. For example, if a task has long duration and large min-tech, then it can be counted as a dominant task. We call this "quantitative" dominance. Moreover, if a task requires expertise such that other tasks in the workload can also be processed by this expertise, then this is "qualitative" dominance. Quantitative dominance is expressed by $MT_j d_j$ and qualitative dominance by $h_j \operatorname{cov}_j$. Our combinatorial algorithm treats tasks in non-increasing order of their weights. The matching weight of a task j is a combined measure of the following criteria: hardness h_j, coverage cov_j, min-tech MT_j, duration d_j, precedence relations, quantitative and qualitative dominance. The weight function in a general form we use is

$$w_j = \mathbf{A}(\overline{h_j} + \overline{\operatorname{cov}_j} + \overline{MT_j} + \overline{d_j}) + \mathbf{B}\overline{MT_j}\,\overline{d_j} + \mathbf{C}\sum_{k:j\to k} w\big(P(k)\big)\overline{MT_k}\,\overline{d_k} + \mathbf{D}\overline{h_j}\,\overline{\operatorname{cov}_j} \tag{15}$$

where "$-$" is used to interpret that all criteria are normalized. Values of coefficients **A**, **B**, **C** and **D** are determined in such a way that contributions of all expressions have the same order of magnitude. In the combinatorial algorithm, a task of high weight is selected with high probability.

5.3 Computing lower bounds

The second part of the preprocessing phase consists of computing lower bounds. We find lower bounds of our problem by solving a simplified problem, which is formulated as a MIP model called the "lower bound model" (LBM). This method was first proposed by Hurkens (2009) and we slightly improved it to obtain tighter lower bounds values for instances with heterogeneous skill distribution. In this section we provide a brief explanation for lower bound model, since this paper focuses more on the contributions in matching models. For further details we refer to Hurkens (2009).

In the simplified problem, skill requirements and precedence relations are relaxed, preemption is allowed, and outsourcing option of tasks is preserved. This problem amounts to finding the minimum time needed to meet the cumulative *man-hour* demand considering technician availabilities. Let $J_p = \{j \in J | P(j) = p\}$ denote the subset of tasks in priority class $p \in P$. The man-hour demand of tasks in J_p can be expressed as

$$MH_{J_p} = \sum_{j\in J_p} MH_j$$

The LBM assumes that priority classes are completed in a pre-specified order. The priority weights ($\{28, 14, 4, 0\}$) given by problem definition suggest the ordering $\{1, 2, 3, 4\}$, however, we consider all priority permutations of practice: $\{1, 2, 3, 4\}$, $\{1, 3, 2, 4\}$, $\{2, 1, 3, 4\}$, $\{2, 3, 1, 4\}$, $\{3, 1, 2, 4\}$, $\{3, 2, 1, 4\}$. Thus lower bound values for all of the mentioned priority permutations are computed due to the fact that any of them may turn out to include the optimum schedule depending on the number of tasks in each priority class. In the schedule construction, priority classes are handled sequentially. The tasks are considered in the order specified by the

priority permutation. The priority permutation with smallest lower bound value is used first to construct schedules, and then the one with second smallest lower bound value and so on. The priority permutations with lower bound values greater than or equal to the lowest cost of constructed schedules are neglected.

5.4 Constructing alternative schedules

In this phase of the algorithm, a number of alternative schedules are constructed by changing several strategies and by varying several parameters.

5.4.1 Initial matching

In the initial matching, we create a bipartite graph in which one partition includes technicians, one partition includes tasks, and an edge is a possible assignment of a certain technician to a certain task (see Fig. 2). On this bipartite graph, the term "matching" denotes a many-to-one type technician-candidate task assignment, and the term "candidate task" denotes a task whose immediate scheduling does not violate feasibility.

A workday schedule is initialized by finding simultaneous assignments of technicians to tasks. Assigning the technicians simultaneously to tasks is superior to assigning them to one task at a time, since the former one has a global view compared to the latter one. The advantage of starting with a global view of skills is more remarkable in the instances with heterogeneous skill distribution.

The problem of finding parallel assignments is formulated as a MIP model, the IMM. The objective of the IMM is to maximize the total matching weight of the selected candidate tasks for initializing teams. Matching weights of tasks represent their power to initialize a team and they are computed as in (15).

Handling large instances: In principle we aim to find the optimal initial matching on the complete set of available technicians and candidate tasks. However, in large instances, one run of the IMM takes longer than the desired time if the number of candidate tasks is high. Therefore, we settle for a heuristic solution by repeatedly applying the IMM to a subset of candidate tasks at a time. The candidate tasks with large matching weights have high priority for being considered in earlier runs. Hence they have high chance to initialize teams.

Table 1 shows the notation, the parameters and the decision variables of the IMM. The mathematical formulation is

$$\text{maximize} \quad \sum_{j \in J'} w_j y_j$$

$$\text{subject to} \quad \sum_{j \in J'} x_{tj} \leq 1 \qquad \forall t \in T' \tag{16}$$

$$\sum_{t \in T'} S_t^{(l,s)} x_{tj} \geq RQ_j^{(l,s)} y_j \quad \forall j \in J', \tag{17}$$

$$\forall (l,s) \in \mathbb{L} \times \mathbb{S}$$

$$y_j, x_{tj} \in \{0,1\} \quad \forall j \in J', \ \forall t \in T'$$

Constraints (16) enforce that a technician can be matched to at most one task. If a task initializes a team, then constraints (17) guarantee that it must be matched to a number of technicians such that its skill requirements are met. Note that a task with a relatively high weight may get more technicians than necessary. Although it may seem an inefficiency, during schedule extension the FMM reallocates technicians dynamically to obtain an efficient packing of tasks. When no technician is left for assigning to tasks or all candidate tasks are assigned to technicians, the initial matching is

Table 1 Sets, indices, parameters, and variables of the IMM

Sets included in the model	
J'	Set of tasks, $J' \subseteq J$
T'	Set of technicians, $T' \subseteq T$
Indices	
j	Task index, $j \in J'$
t	Technician index, $t \in T'$
Parameters	
w_j	Weight of task j
$S_t^{(l,s)}$	Competence of technician t in domain s at level l
$RQ_j^{(l,s)}$	Number of skilled technicians required by task j in domain s at level l
Variables	
x_{tj}	Binary variable indicating whether technician t is assigned to task j
y_j	Binary variable indicating whether task j initializes a team

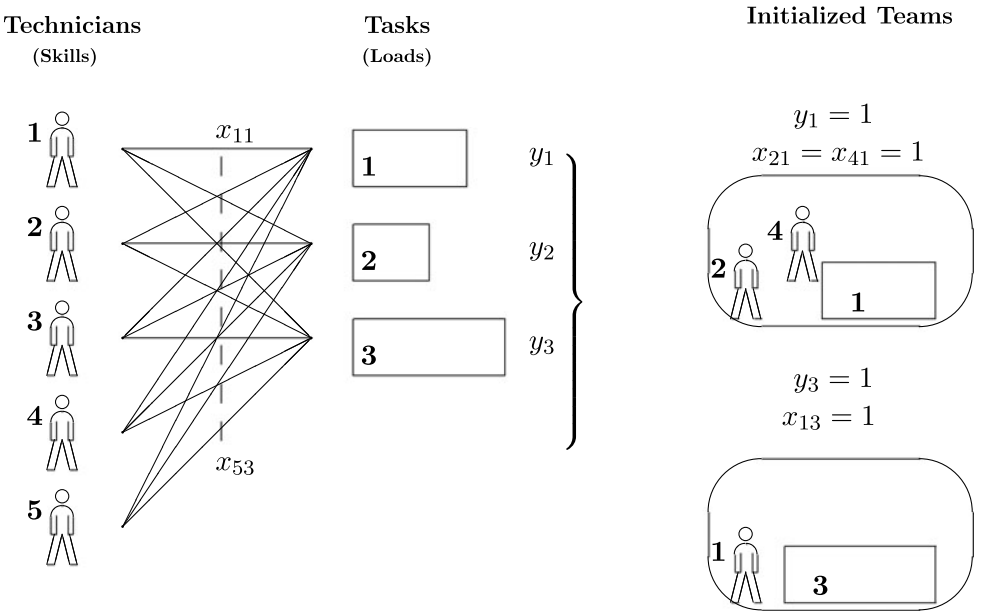

Fig. 2 Initializations of teams in the IMM

completed. The initialized workday schedule has teams with single tasks in their workloads.

Figure 2 illustrates team initializations with an example of five technicians and three tasks. In the solution of the IMM, it turns out that two teams are initialized by j_1 and j_3. In the partial schedule team τ_1 (τ_2) has load with duration d_1 (d_3). The initialized teams have the technicians $T(\tau_1) = \{t_4, t_2\}$ and $T(\tau_2) = \{t_1\}$.

5.4.2 *Extending the partial day schedule*

Once the teams are initialized in the initial matching, the partial workday schedule is extended by iteratively inserting more tasks. Adding more tasks to the schedule is not trivial, since some tasks may require slightly or even completely different skills than the ones already scheduled. Therefore, the easier reallocation of technicians among teams, the more candidated tasks that may be inserted into workday schedule. This is the flexibility of our approach. In the FMM, reallocation of technicians among teams is achieved by letting them be *conditionally* available.

Generalization from "technicians and tasks" to "skills and loads": In a partially constructed workday schedule, a team has two features, its technicians and its workload. First, a team can be perceived as a combination of *skills* if technicians are considered. Second, a team can also be perceived as a *load* if its workload is considered. Therefore from the first (second) point of view a team represents a *skill* (*load*). Moreover unassigned individual technicians (candidate tasks) can be perceived as *skills* (*loads*). These observations lead us to consider every item of the problem either as "skill" or as "load." Table 2 shows types of skills and loads. Instead of technicians and tasks we will have skills and loads in the matching model to extend the partial day schedule.

Table 2 Types of skills and loads

Skills	
Active	Total technician skills in a team
Passive	Unassigned technician
Latent	Conditionally available technician
Loads	
Active	Workload of a team
Passive	Candidate task

The skills and the loads of teams are called "active," since they are actively comprising the partial workday schedule. The technicians who have not been assigned yet are called "passive" skills. Similarly, the candidate tasks are called "passive" loads. Individual technicians in the teams are called "latent" skills. Note that a team has one active skill that the collective skill of technicians, whereas the number of latent skills of a team is equal to the number of that team's technicians.

In a matching, if the active load of a team is matched with a new combination of skills, then team's latent skills, not included in the new skill combination, become free to be assigned to other loads. Thus latent skills are *conditionally* available for matchings and they play an important role in scheduling candidate tasks (passive loads) by determining skill combinations of teamloads flexibly.

Skill-load assignments are found by defining a bipartite graph in which the left partition includes skills and the right partition includes loads. In this bipartite graph, many-to-one type matching consists of skill-load assignments. Teams are represented in the left partition by their active skills and latent skills, and in the right partition by their active loads. In

addition we have (passive) skills of unassigned technicians in the left partition and (passive) loads of candidate tasks in the right partition. The possible extension cases that can appear in a matching solution are listed below:

Case 1 an active skill is matched with:
- (a) an active load: *merging*
- (b) a passive load: *extending* team load

Case 2 an active load is matched with:
- (a) passive and/or latent skills: *recombining* technicians
- (b) skills including an active skill: *merging*

Case 3 a passive load is matched with:
- (a) passive and/or latent skills: *initializing* a team
- (b) skills including an active skill: *extending* team load

In the FMM, a candidate task can be matched with "*any*" combination of skills provided that every teamload in the model either keeps its skills or finds a new skill combination to stay being processed. This is the key aspect of our matching model resulting in high flexibility. Each technician, no matter in a team or not, becomes a potential skill for candidate tasks. Candidate tasks may be added to the partial schedule by joining a teamload (Case 1a) or by initializing a team (Case 3a). While joining to a teamload, a candidate task may bring some additional technicians to the team, if necessary.

The condition for a latent skill to be assigned to another load is that the active load of its team is matched with new skill combination missing it. In other words, assigning an active load with new skills leads to an opportunity of using some of its technicians in other matchings. Generally speaking, matching of active loads to skill combinations is triggered by passive loads (candidate tasks), since they mainly contribute to the objective of matching model. The higher weight a candidate task has, the more power to force the current partial schedule to find the needed skill combination. In light of this fact we have the following observation:

Observation *A candidate task with sufficiently high weight may force teams to merge or recombine their technicians, thereby making an expert technician available.*

Sequencing decision in extending teamloads: In Case 1b and Case 3b, a teamload is extended by adding the matched candidate task. Let j be the candidate task matched to the active skill of team τ. The start time of j is determined by a procedure in which already scheduled tasks are considered starting with the last one in the sequence. Let j' be a scheduled task under consideration. The following conditions are checked:

$$w\big(P(j')\big) > w\big(P(j)\big) \tag{18}$$

$$w\big(P(j')\big) = w\big(P(j)\big) \quad \text{and}$$
$$\big|\{k \in J \,|\, j' \in Pred(k)\}\big| > \big|\{k \in J \,|\, j \in Pred(k)\}\big| \tag{19}$$

$$\big|\{k \in J \,|\, j' \in Pred(k) \text{ and } CT_k - d_k < CT_{j'} + d_j\}\big| > 0 \tag{20}$$

Condition (18) is the case in which the priority class of j' has more weight than the one j belongs to. In (19), both tasks j and j' belong to the same priority class and j' has more successors than j. Finally, in (20), j' has a scheduled successor which starts earlier than $CT_{j'} + d_j$. If none of the cases in (18), (19), and (20) is true, then j is allowed to start at the start time of j' and the start time of j' increases by d_j. Next, the task preceding j' in the sequence is considered and the same checking is done for that task as well. This checking procedure continues until either a task that does not let j start at its start time is encountered, or j starts first in the sequence. Start times of all tasks in the sequence are determined accordingly.

Sequencing decision in merging two matched teamloads: If an active skill and an active load are matched, then the tasks in the matched active load and tasks in the workload of the matched team are merged in a way that the relative ordering of tasks in both loads stay the same. However, the order between tasks from different teamloads is determined by checking lexicographically first, priority classes, and second, number of successors. Obviously, the tasks in priority classes with higher weights and having more successors start earlier.

Merging teamloads generally leads to an increase in overall efficiency of the partial schedule. For example, let two tasks, say j_1 and j_2, with the same skill requirements have initialized different teams in the initial matching. If the active load of j_1 is matched to the active skills of j_2 in a solution of the FMM, both tasks are processed by the technicians assigned to j_2 and the technicians who were previously assigned to j_1 contribute other matchings in the same solution.

Figures 3 and 4 illustrate examples of extending the partial schedule of the example in Fig. 2. In order to show all possible cases, we illustrated two different scenarios corresponding to two different solutions obtained with different parameter settings.

In Fig. 3, also in Fig. 4, the FMM includes 2 teams (τ_1, τ_2) with technicians $T(\tau_1) = \{t_4, t_2\}$ and $T(\tau_2) = \{t_1\}$, 2 unassigned technicians (t_3, t_5) and one candidate task (j_2). Latent skills are shown in black boxes and linked to their current teams. The first two vertices of (left) right partition are active (skills) loads. There is no edge drawn between an active skill and an active load of the same team, since this matching does not make any sense and it is forbidden in the model.

In scenario 1 (Fig. 3), the active load of τ_2 is matched to a new skill combination including the active skill of τ_1 and passive skill of t_5. This matching is an example of merging (Case 1a) and the combined load is performed by the

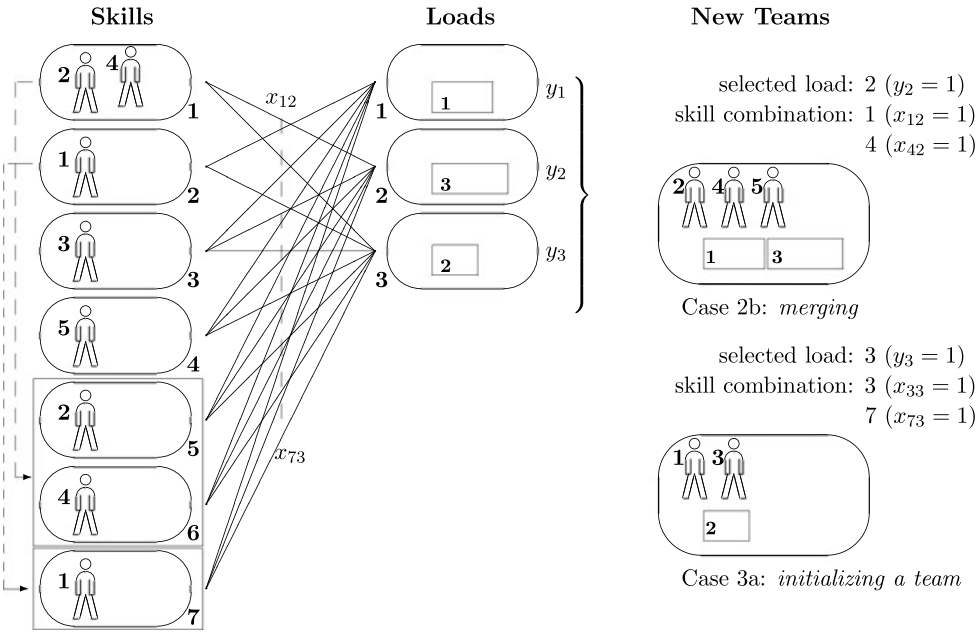

Fig. 3 Extending partial schedule with the FMM (Scenario 1)

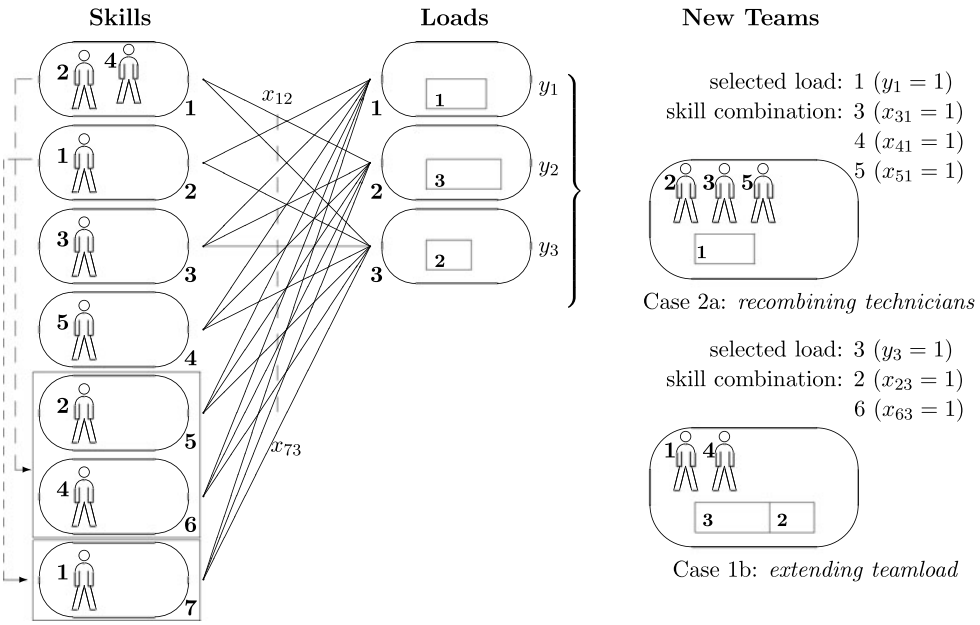

Fig. 4 Extending partial schedule with the FMM (Scenario 2)

newly matched skill combination. Matching active load of τ_2 allowed t_1 to be in the initialized team of j_2 (Case 3a) together with technician t_3.

In scenario 2 (Fig. 4), τ_1 recombined its technicians (Case 2a) by being matched to passive skills of unassigned technicians t_3, t_5 and latent skill of technician t_2 who was in the previous combination as well. The number of technicians of τ_1 seems increased by one and this increase can be explained as an adjustment to have technician t_4 in the extended team τ_2 (Case 1b) as we now have j_2 in one load.

Due to its mentioned flexibility aspects, we call the bipartite matching model the *flexible matching model*, or the FMM. The main contribution of this study is introducing this model. Hurkens (2009) extends the partial day schedule using two different matching models iteratively, where the first one is used to assign multiple tasks simultaneously, and the second one is used to find efficient recombination of technicians. The FMM carries out these two steps simultaneously.

Forbidden assignments: In the FMM some assignments may cause infeasibility and some are useless. The edges on the bipartite graph corresponding to these assignments are detected in advance and they are not allowed to be in any matching. These edges can be listed as follows:

– the edges between active load and active skill of the same team,
– the edges that lead to a total duration of a teamload longer than a work day
– the edges that result in violation of a precedence relation
– the edges that cause a delay in an already completed priority class

Punished assignments: The number of teams included in the FMM is limited. Therefore an assignment, although it seems profitable on the current bipartite graph, may not be so for the complete partial workday schedule. The immediate example is a matching of active skill and passive load in which an active skill (technician group of a team) has more technicians than needed for the passive load (candidate task). The FMM has penalty variables to detect and prevent these assignments.

5.4.3 Mathematical formulation of the FMM

The FMM finds many-to-one type assignments on a bipartite matching model whose left (right) partition represents *skills* (*loads*). The sets, indices, parameters and variables are listed in Table 3.

Note that $|\mathbb{T}| = |\mathbb{W}| = |\mathbb{M}|$, since each team has its skill in $\mathbb{T}$ and its load in $\mathbb{W}$ with corresponding indices $(\sigma, \lambda) \in \mathbb{M}$.

$$\text{maximize} \quad \sum_{\lambda} w_\lambda y_\lambda - z_\lambda$$

$$\text{subject to} \quad \sum_{\lambda} x_{\sigma\lambda} \le 1, \qquad \forall \sigma \in \Delta \tag{21}$$

$$\sum_{\sigma \in \mathbb{T}} x_{\sigma\lambda} \le 1, \qquad \forall \lambda \in \Lambda \tag{22}$$

$$x_{\sigma\lambda} = 0, \qquad \forall (\sigma, \lambda) \in \mathbb{F} \tag{23}$$

$$\sum_{\lambda'} x_{\sigma\lambda'} + y_\lambda \le 1, \quad \forall (\sigma, \lambda) \in \mathbb{M} \tag{24}$$

$$\sum_{\sigma' \in T(\sigma)} \sum_{\lambda'} x_{\sigma'\lambda'} \le \big|T(\sigma)\big| y_\lambda, \qquad \forall (\sigma, \lambda) \in \mathbb{M} \tag{25}$$

$$\sum_{\sigma} S_\sigma^{(l,s)} x_{\sigma\lambda} \ge RQ_\lambda^{(l,s)} y_\lambda, \quad \forall (l, s) \in \mathbb{L} \times \mathbb{S}, \qquad \forall \lambda \in \Lambda \tag{26}$$

$$\sum_{\sigma \in \mathbb{T}} \big|T(\sigma)\big| x_{\sigma\lambda} + \sum_{\sigma \in \Delta \setminus \mathbb{T}} x_{\sigma\lambda} - mt_\lambda \le z_\lambda, \qquad \forall \lambda \in \Lambda \tag{27}$$

$$x_{\sigma\lambda}, y_\lambda \in \{0, 1\}, \qquad z_\lambda \ge 0$$

A skill can be assigned to at most one load (constraints (21)). A load may be matched to at most one active skill

Table 3 Sets, indices, parameters and variables of the FMM

Sets included in the model	
Δ	Set of all skills in the left partition
Λ	Set of all loads in the right partition
$\mathbb{T}$	Set of active skills
$\mathbb{W}$	Set of team loads
$\mathbb{F}$	Forbidden edges, $\mathbb{F} \subseteq \Delta \times \Lambda$
$\mathbb{M}$	Set of edges between skill and load of the same team
Indices	
σ, σ'	Skill index, $\sigma, \sigma' \in \Delta$
λ, λ'	Load index, $\lambda, \lambda' \in \Lambda$
Parameters	
$T(\sigma)$	Set of latent skills of active skill σ
w_λ	Weight associated to load λ, equal to w_j if $\lambda \notin \mathbb{W}$ where j is candidate task and otherwise equal to 0
mt_λ	The number of technicians used for performing all tasks in load λ
$S_\sigma^{(l,s)}$	Skill value of σ in domain s at level l
$RQ_\lambda^{(l,s)}$	Number of skilled technicians required by λ in domain s at level l
Variables	
$x_{\sigma\lambda}$	Binary variable indicating whether σ is assigned to λ
y_λ	Binary variable indicating whether λ assigned to a skill combination
z_λ	Inefficiency penalty of λ

according to constraints (22). It can be matched to any number and any combination of passive and latent skills though. Having constructed the set $\mathbb{F}$ in advance, edges in $\mathbb{F}$ are forbidden by constraints (23).

In the FMM, a team has an active skill $\sigma \in \mathbb{T}$ and latent skills $T(\sigma)$ in the left partition and has a load $\lambda \in \mathbb{W}$ in the right partition. According to constraints (24), a team can contribute to extending the day schedule in one of the following ways: either its active skill is matched to a load or its active load is matched to skills. In the former case, the matched load is added to team's workload and technicians of the team stay together. Some additional technicians may join to team as well, if some other latent or passive skills are also assigned to matched load. In the latter case, a skill combination is assigned to active load (constraints (26)) and latent skills of the team may be used in other matchings (constraints (25)). Inefficiency variables z_λ try to avoid the assignments that have an unnecessarily high number of technicians (constraints (27)). Here $mt_\lambda = MT_j$ for a passive load λ with candidate task j and $mt_\lambda = |T(\sigma)|$ for an active load λ currently assigned to σ. For instance, if a load of a candidate task with $MT = 2$ is assigned to a team skill of 5 technicians, then this assignment is penalized by constraints (27) on the value of z_λ. Note that inefficiency is not forbidden in the FMM, but discouraged by penalizing.

The objective function is the sum of the weights of selected loads and the inefficiency drop. The candidate tasks contribute the objective by their weights and influence the allocation of skills among teams. Note that a task with sufficiently high weight can even cause some inefficiency to get scheduled.

5.4.4 *Strategies applied in constructing alternative schedules*

In this section we explain the strategies applied to find alternative schedules. We have observed in experiments that each of those strategies may lead to a best solution.

Efficiency in initial matching: When the formulation of the FMM is carefully examined, it is not difficult to see that if $\mathbb{T} = \emptyset$, the FMM boils down to the IMM plus efficiency constraints. So one of our strategies is adding efficiency constraints to the Initial Matching. This strategy is especially beneficial in instances where the average duration of tasks is close to workday length.

Fully loaded teams: While extending the partial day schedule some teams may reach a workload with total duration of a workday. As a strategy, we include a few of those fully loaded teams in the FMM in the hope to lower their skills excess by making some of their technicians conditionally available for candidate tasks. Fully loaded teams can contribute to extending partial schedule by recombining their technicians.

Number of teams in the FMM: In order to obtain the optimal solution of the FMM in a reasonably short time, a fixed number of teams are included. So we have $|\mathbb{T}| = |\mathbb{W}| \in \{3, 4, 5, 6, 7, 8\}$. As long as the time limit allows, we construct schedules for each fixed number. In large instances usually a few of them can be used.

Selection of candidate tasks: Candidate tasks are included in the FMM in sequential order according to their priority classes. As a strategy we allow some of the special tasks in succeeding priority classes to be in the FMM. This way we aim to avoid the delays of the priority makespan due to rare expertise.

6 Computational results

6.1 On the rare expertise in problem instances

Three sets of problem instances were provided by France Telecom in the 2007 ROADEF Challenge. The descriptive statistics of the instances can be seen in Table 4. In the instance sets, the number of skill domains (levels) varies from 3 to 40 (2 to 7). Data set A was released in the first stage of the challenge for participants to start implementing their solution approaches. Data set B was released for fine tuning and the data set X was used for final evaluation. Instances in the set A are smaller than the instances of set B and set X. There is no significant difference between data set B and X in terms of number of technicians and number of tasks. However the number of special tasks is the point where they differ. (See Sect. 5.2.2 for a definition of special task.)

The costs of schedules constructed by the FMM and other heuristics are given in Table 5. We used the time limit 20 minutes, as specified in the 2007 ROADEF Challenge and all results are obtained on a laptop with Intel Core 2 Duo 1.6 GHz Processor, 4 GB RAM. The MIP models, the IMM and the FMM, are solved using the solver CPLEX 12.1.0. Table 6 shows the number of schedules constructed for each problem instance and the time needed to construct these schedules. The first column in Table 5 shows the problem instances and the next four columns report the results found by the FMM, Hurkens (2009), Cordeau et al. (2010) and Estellon et al. (2009). In each of these columns, the first entries are schedule costs and the second entries are the relative difference in percentage that is defined as the difference of a schedule cost to the best schedule cost ever found (best schedule costs are listed in the column with title "BEST"). In the column "BEST," we report the lowest schedule costs by considering the results of the 2007 ROADEF Challenge as well. The last column, labeled "LB," lists lower bound values of instances.

It is seen in the last column of Table 4 that data set X includes instances with a higher number of special tasks. The

Table 4 Problem instances A, B and X

Instance	Data set A					Data set B					Data set X				
	$\|T\|$	$\|J\|$	$\|S\|$	$\|L\|$	$\|SP\|$	$\|T\|$	$\|J\|$	$\|S\|$	$\|L\|$	$\|SP\|$	$\|T\|$	$\|J\|$	$\|S\|$	$\|L\|$	$\|SP\|$
1	5	5	3	2	0	20	200	4	4	0	60	600	15	4	27
2	5	5	3	2	0	30	300	5	3	0	100	800	6	6	0
3	7	20	3	2	1	40	400	4	4	0	50	300	20	3	0
4	7	20	4	3	0	30	400	40	3	15	70	800	15	7	0
5	10	50	3	2	1	50	500	7	4	9	60	600	15	4	13
6	10	50	5	4	5	30	500	8	3	0	20	200	6	6	3
7	20	100	5	4	1	100	500	10	5	0	50	300	20	3	0
8	20	100	5	4	0	150	800	10	4	0	30	100	15	7	5
9	20	100	5	4	3	60	120	5	5	0	50	500	15	4	10
10	15	100	5	4	2	40	120	5	5	0	40	500	15	4	14

SP: Special Tasks, $SP \subseteq J$

average number of special tasks of the instance groups A, B and X are 1.3, 2.4 and 7.2, respectively. This hints that data set X instances have a heterogeneous skill distribution among technicians and therefore rare expertise is observed. The challenge in the instances of rare expertise can be realized by checking the gap between the best found schedules and lower bounds. In Table 5 we see that the gap between best schedules and lower bounds is smaller in data sets A and B compared to data set X. This may show either the weakness of lower bounds or the case that the approaches are not successful in handling rare expertise or both.

In the final evaluation of 2007 ROADEF Challenge, seven best schedule costs (out of 10 instances in set X) were due to Hurkens (2009). This shows that the solution approach of Hurkens (2009) was promising for cases of rare expertise. If the column BEST is examined in Table 5, it is seen that the FMM found eight best schedules in data set X. In our opinion, this is the result of the flexibility of our matching model. The flexibility leads to more efficient packing of special tasks and higher utilization of the rare expertise. In the instances X9 and X10, the gap between best schedule and lower bound has been decreased remarkably. As an improved version, the FMM found better schedules in all X instances compared to Hurkens' results.

If all data sets are considered, it seems reasonable to conclude that data sets B and X are better representatives of the real case instances due to the high number of technicians, tasks and skill domains. Moreover rare expertise is also a situation companies encounter in their operations. In instances with a small number of technicians, tasks and skill domains, schedule costs are sensitive to individual assignments, whereas in large instances the number of feasible schedules are high and the schedules are not sensitive to individual assignments. Therefore in our opinion, large instances are better to test the reliability of the algorithms.

6.2 On the performances of heuristics

It is remarkable that the FMM has an average difference with the best available solution of 0.9% in data set X, whereas it is 7.1%, 13.7% and 15.8% for Hurkens (2009), Cordeau et al. (2010) and Estellon et al. (2009), respectively. The results of Cordeau et al. (2010) are the best ones in data set A, however their average distance increases from A to B and from B to X. Hurkens (2009) has an increase from A to B and stays almost at the same level from B to X. The FMM starts with a high average distance in data set A and draws a slight increase from A to B. In sets B and X, it performs remarkably well. The superior performance in set X shows that in case of rare expertise, the FMM can find compact schedules and experiences less skill excess in assignments. Particularly, the instances X1, X5, X9 and X10 are sensitive to expert availabilities due to the high number of special tasks. Estellon et al. (2009) also underline these instances and emphasize the gap between the combinatorial approach by Hurkens (2009) and the other solution approaches.

7 Conclusions and future research

In this paper, we proposed a solution methodology that uses a flexible matching model as a core engine for a special multi-skill workforce scheduling problem. The scheduling problem was defined by France Telecom in the 2007 ROADEF Challenge. The opportunity of outsourcing some tasks is one aspect of our problem that distinguishes it from the similar ones defined in the literature. Technicians must work in teams for a workday and it is assumed that they can simultaneously use their skills in all domains while performing tasks.

The main contribution of this study is introducing flexibility in the matching model that was firstly proposed by

Table 5 Results of problem instances A, B and X

Instance	FMM (%)		Hurkens (%)		Cordeau (%)		EsGaNo (%)		BEST*	LB
A1	2340	0.0	2340	0.0	2340	0.0	2340	0.0	2340	2310
A2	4755	0.0	4755	0.0	4755	0.0	4755	0.0	4755	2100
A3	11880	0.0	11880	0.0	11880	0.0	11880	0.0	11880	11340
A4	13452	0.0	13620	1.2	13452	0.0	14040	4.4	13452	10680
A5	29355	1.8	29355	1.8	29355	1.8	29400	1.9	28845	26940
A6	20055	6.7	20280	7.9	18795	0.0	18795	0.0	18795	17640
A7	30960	1.4	32520	6.5	30540	0.0	30540	0.0	30540	28672
A8	17355	2.6	18960	12.1	17700	4.6	20100	18.8	16920	16216
A9	28280	3.4	28320	3.6	27692	1.3	27440	0.3	27348	25558
A10	39300	2.6	40650	6.1	38636	0.9	38460	0.4	38296	36992
Average		*1.8*		*3.9*		*0.9*		*2.6*		
B1	34575	2.0	35460	4.6	37200	9.7	33900	0.0	33900	31875
B2	16755	5.6	18300	15.3	17070	7.6	16260	2.5	15870	14280
B3	16275	1.7	16965	6.0	18015	12.6	16005	0.0	16005	13965
B4	23925	0.6	27015	13.6	23775	0.0	24330	2.3	23775	16800
B5	88920	0.3	94200	6.2	117540	32.5	88680	0.0	88680	79530
B6	28785	5.1	30510	11.4	27390	0.0	27675	1.0	26955	24180
B7	31620	0.0	33060	4.6	33900	7.2	36900	16.7	31620	25290
B8	35520	10.4	32160	0.0	33240	3.4	36840	14.6	32160	31890
B9	28080	0.0	28080	0.0	29760	6.0	32700	16.5	28080	25680
B10	35040	1.0	35040	1.0	35640	1.7	41280	19.0	34680	32370
Average		*2.7*		*6.2*		*8.1*		*7.3*		
X1	146220	0.0	151980	3.9	159300	8.9	180240	23.3	146220	136680
X2	7740	6.6	9090	25.2	8280	14.0	8370	15.3	7260	5700
X3	48720	0.0	50400	3.4	50400	3.4	50760	4.2	48720	36060
X4	64600	0.0	65640	1.6	66780	3.4	68960	6.7	64600	58230
X5	144750	0.0	147000	1.6	157800	9.0	178560	23.4	144750	130995
X6	9690	2.2	10440	10.1	9900	4.4	10440	10.1	9480	6150
X7	32040	0.0	33120	3.4	47760	49.1	38400	19.9	32040	25410
X8	23220	0.0	23580	1.6	24060	3.6	23800	2.5	23220	17600
X9	122700	0.0	136020	10.9	152400	24.2	154920	26.3	122700	98805
X10	120300	0.0	131700	9.5	140520	16.8	152280	26.6	120300	87210
Average		*0.9*		*7.1*		*13.7*		*15.8*		
Overall		***1.8***		***5.8***		***7.6***		***8.6***		

* Results of the 2007 ROADEF Challenge are also considered

Hurkens (2009). Moreover, we propose several key measurements for tasks. The flexibility of our matching model resulted in better packing of the expert-requiring tasks especially in instances where skill distribution among technicians is heterogenous. Besides the rare-expertise instances, our results are also remarkably superior in large instances.

The flexibility in the proposed matching model allows to construct a high number of schedules. As can be seen in its mathematical formulation, weights of tasks are important parameters to obtain good quality solutions. There is no unique measurement for the weight of a task, but we proposed a measurement to include rare expertise concerns. We believe that our matching models are proper tools to test the success of these measurements. As a topic of further research, it will be useful to adapt our matching model to other multi-skill workforce scheduling problems with necessary modifications. We are recently working on adapting our approach to the multi-skill workforce scheduling problem studied by Bellenguez and Neron (2004). We aim to get good quality solutions for this problem in short times.

Table 6 Running times and number of constructed schedules

Instance	Data set A		Data set B		Data set X	
	Time (sec.)	#Schedules	Time (sec.)	#Schedules	Time (sec.)	#Schedule
1	9	745	1086	125	1110	9
2	10	830	1090	105	1084	168
3	1085	9108	1087	28	1091	136
4	1079	7563	1106	35	1096	9
5	1081	2209	1089	130	1173	8
6	1081	2617	1086	54	1081	560
7	1082	838	1109	45	1086	164
8	1085	693	1142	3	1087	157
9	1083	793	1090	328	1173	14
10	1085	910	1081	325	1110	14

Acknowledgements This research is supported by France Telecom/TUE Research agreement No. 46145963. We are grateful to Alexandre Laugier and Anne-Marie Bustos for their cooperation and help in testing various versions of the algorithm. Thanks are also due to Gerhard J. Woeginger for helpful discussions on the interpretation of MIP models. Finally, we thank the reviewers for their comments, which helped a lot in improving this paper.

References

Avramidis, N. A., Chan, W., Gendreau, M., L'Ecuyer, P., & Pisacane, O. (2010). Optimizing daily agent scheduling in a multi-skill call center. *European Journal of Operational Research*, *200*(3), 822–832.

Ballou, D., & Tayi, G. (1996). A decision aid for the selection and scheduling of software maintenance projects. *IEEE Transactions on Systems Man and Cybernetics Part A Systems and Humans*, *26*(2), 203–212.

Bellenguez, M. O., Canon, C., & Neron, E. (2005). Ordonnancement des formations des tele-oprateurs dans un centre de contacts clients. In *Proc. ROADEF 2005*, Tours, France.

Bellenguez, M. O. (2006). Methods to solve multi-skill project scheduling problem. PhD thesis, Francois Rabelais University, Tours, France.

Bellenguez, M. O., & Neron, E. (2004). Lower bounds for the multi-skill project scheduling problem with hierarchical levels of skills. In *Proceedings of Practice and Theory of Automated Timetabling (PATAT2004)*, Pittsburgh, PA, USA (pp. 429–432).

Bellenguez, M. O., & Neron, E. (2007). A branch-and-bound method for solving multi-skill project scheduling problem. *RAIRO Operations Research*, *41*, 155–170.

Brucker, P., Drexl, A., Möhring, R., Neumann, K., & Pesch, E. (1999). Resource-constrained project scheduling: notation, classification, models, and methods. *European Journal of Operational Research*, *112*(1), 3–41.

Cai, X., & Li, K. N. (2000). A genetic algorithm for scheduling staff of mixed skills under multi-criteria. *European Journal of Operational Research*, *125*(2), 359–369.

Cordeau, J. F., Laporte, G., Pasin, F., & Ropke, S. (2010). Scheduling technicians and tasks in a telecommunication company. *Journal of Scheduling*, *13*(4), 393–409.

De Reyck, B., & Herroelen, W. S. (1999). The multi-mode resource-constrained project scheduling problem with generalized precedence relations. *European Journal of Operational Research*, *119*(2), 538–556.

Drezet, L.-E., & Billaut, J.-C. (2008). A project scheduling problem with labour constraints and time-dependent activities requirements. *International Journal of Production Economics*, *112*, 217–225.

Dutot, P., Laugier, A., & Bustos, A. (2006). Technicians and interventions scheduling for telecommunications. In *Problem description of ROADEF 2007 challenge*.

Estellon, B., Gardi, F., & Nouioua, K. (2009). *High-performance local search for task scheduling with human resource allocation*. *Lecture notes in computer science* (vol. 5752, pp. 1–15).

Garey, M. R., & Johnson, D. S. (1979). *Computers and intractability: a guide to the theory of NP-completeness*. San Francisco: Freeman (p. 221).

Gutjahr, W. J., Katzensteiner, S., Reiter, P., Stummer, C., & Denk, M. (2008). Competence-driven project portfolio selection, scheduling and staff assignment. *Central European Journal of Operations Research*, *16*(3), 281–306.

Hartmann, S., & Briskorn, D. (2010). A survey of variants and extensions of the resource-constrained project scheduling problem. *Central European Journal of Operations Research*, *207*, 1–14.

Heimerl, C., & Kolisch, R. (2010). Scheduling and staffing multiple projects with a multi-skilled workforce. *OR Spectrum*, *32*(2), 343–368.

Hurkens, C. A. J. (2009). Incorporating the strength of MIP modeling in schedule construction. *RAIRO Operations Research*, *43*, 409–420.

Li, H., & Womer, K. (2009). Scheduling projects with multi-skilled personnel by a hybrid MILP/CP benders decomposition algorithm. *Journal of Scheduling*, *12*(3), 281–298.

Valls, V., Perez, A., & Quintanilla, S. (2009). Skill workforce scheduling in service centers. *European Journal of Operational Research*, *193*(3), 791–804.

Wu, M. C., & Sun, S. H. (2006). A project scheduling and staff assignment model considering learning effect. *The International Journal of Advanced Manufacturing Technology*, *28*, 1190–1195.

Yoshimura, M., Fujimi, Y., Izui, K., & Nishiwaki, S. (2006). Decision-making support system for human resource allocation in product development projects. *International Journal of Production Research*, *44*(5), 831–848.