A heuristic method to schedule training programs for SMEs

Mahmood Rezaei[1,*], Fahimeh Shamsaei[2], Iman Mohammadian[3], Mathieu Van Vyve[2]

[1]*HEC Management School, University of Liège, Belgium*

[2]*Center for Operations Research and Econometrics, Université Catholique de Louvain, Belgium*

[3]*Iran University of Science and Technology, Tehran, Iran*

Abstract

During the life period of Small and Medium Enterprises (SMEs) in incubators they need some training programs to acquire the required knowledge in order to survive and succeed in the business environment. This paper presents a heuristic method based on an optimization model to schedule these programs at the most suitable times. Based on the proposed heuristic, each training program is implemented in a suitable time by considering the SMEs' requirements and some other logical constraints. The proposed heuristic is described in detail, and its implementation is demonstrated via a real-life numerical example. The numerical results of the heuristic are compared with other methods.

Keywords: course scheduling; heuristics; incubators; job scheduling; small and medium enterprises.

1. Introduction

Incubators have become an integral part of business in many countries, and governors regard them as a crucial tool in the economic development. The role of Small and Medium Enterprises (SMEs) is not negligible in the boom of every economy, and incubators are one of most effective environments to support SMEs to develop their innovations, commercialize their research results, and achieve a sustainable growth. This support ranges from providing office space and physical equipments with low cost to assistance in developing business plan, capital acquisition, and other specialized services with high added-value (Bergek and Norman, 2008; Grimaldi and Grandi, 2005). The life period of SMEs in an incubator usually takes 24-36 months. After this incubating period, they are expected to become independent with a sustainable development in their business. The early years of the life of SMEs in incubators have been the focus of considerable research effort. Specially, identification of the important factors and conditions that expedite their growth process and success has attracted much attention from the research community. Today's competitive world has diversified the required skills

* Corresponding author. Tel.: +32-42327342

E-mail addresses: m.rezaei@ulg.ac.be (M. Rezaei), fahimeh.shamsaei@student.uclouvain.be (F. Shamsaei), iman.mohammadian@gmail.com (I. Mohammadian), mathieu.vanvyve@uclouvain.be (M. Van Vyve).

and knowledge to succeed in business. Whereas universities usually provide the preliminary skills from the theoretical point of view, incubators are specialized environments to support SMEs' leaders by providing other necessary skills and knowledge. Consequently, there exists a gap between the required knowledge and the possessed knowledge by entrepreneurs (Chrisman and McMullen, 2004). The knowledge and skills of SMEs' leaders is very important to increase their performance, competitiveness, and survival chance in the business environment, and it has been argued that SMEs' failures are typically due to the lack of such managerial skills (Feeser and Willard, 1990; Martocchio and Baldwin, 1997; Zahra and Covin, 1993). For any SME to be, and to remain, successful over a long period of time there must be the capability to adapt to new circumstances, especially in the early years of life in an incubator. During the past decade, SMEs have been encouraged to utilize the services of knowledge centers. In spite of these efforts there are still some questions about the nature and the extent of SMEs' knowledge requirements and the efficient ways to transfer this knowledge. However, there is a common agreement on the necessity of continuous training to update and enhance the required skills of employees, and even leaders, in all organizations including SMEs (Salas and Cannon-Bowers, 2001). Implementation of the training programs for SMEs is one of the main duties of incubators (Aaboen, 2009), but it is not a trivial task to choose the right training programs at the right times (Banfield et al., 1996). Our literature survey failed to find many papers directly related to schedule training programs for SMEs. The most relevant research area in the literature is course scheduling, which has attracted the attention of many researchers in the domain of operational research for many years. Course scheduling problems aim to determine which courses should be taught on which days, on what times, and in which rooms, subject to some side constraints (Thompson, 2005). There exist many papers related to the course scheduling problem, and each paper focuses on some specific aspects of this problem. However, the majority of the existing papers have been devoted to course scheduling at universities, where professors are assigned to courses and courses are assigned to time slots and classrooms. Different objective functions might be considered such as conflict in teaching hours, professors' preferences, and continuity of teaching hours. Usually, two types of constraints are defined: those which must be strictly satisfied under any circumstances (hard constraints) and those which are not necessarily satisfied but whose violations should be minimized (soft constraints). Any solution which satisfies all the hard constraints is called a feasible solution. Due to the complexity of the real-world course scheduling problem, the soft constraints may need to be relaxed since it is not usually possible to generate solutions without violating some of them. Soft constraints are usually used within the cost evaluation function to evaluate how good the solutions are (Lu and Hao, 2010). Some exact techniques have been proposed for solving the course scheduling problem, including constraint programming (Valouxis and Efthymois, 2003) and integer programming (Daskalaki and Birbas, 2005). Course scheduling problem is known as an NP-Complete problem. Since NP-Complete problems cannot be solved efficiently as the size of the problem increases (Head and Shaban, 2007) heuristic approaches (Dimopoulou and

Miliotis, 2001) as well as meta-heuristics such as ant colony optimization (Azimi, 2005), genetic algorithms (Wang, 2003), tabu search (Alvarez-Valdes et al., 2002), and simulated annealing (Zhang et al., 2010) have been extensively applied by researchers to find a near-optimal solution. Although various course scheduling problems and approaches have been investigated in the literature, no specific one can be applied universally due to the specific features of each problem. Some recent works focus to bridge the gap between research and the real world applications (Zhang et al., 2010). In practice, course scheduling is not restricted to universities; many organizations use course scheduling techniques to schedule the required training programs for their employees. These training programs are crucial elements of human resource development in order to follow a sustainable development. They increase the quality of human resources as well as the organizational long term efficiency and productivity. The impact of training programs can be significantly enhanced if they are appropriately scheduled (Juang et al., 2007). Studies conducted in 2006 show that in this year, organizations in the United States spent a total of $55.8 billion on training. However, in spite of such enormous investment in training, there was little evidence of verifiable return or of the effective results (Tharenou et al., 2007). In view of the experts, the main reason of this failure is the lack of needs assessment before implementing the training programs or inappropriate times to implement them. So, both careful *needs assessment* and *in-time implementation* of training programs are necessary to increase the efficiency of the training programs. In other words, the training programs should be implemented not only with a high quality, but also at a suitable time and based on needs assessment. This paper proposes a heuristic method based on an optimization model to specify the most suitable time for implementation of each training program. The remaining of this paper is organized as follows. The optimization model to schedule training programs is presented in Section 2. Then, in Section 3 the proposed heuristic method to find a solution close to the optimal solution of the optimization model is discussed in detail. Section 4 explains an equivalent job scheduling problem in order to show that the application of our problem is not merely restricted to SMEs. A real-life numerical example is provided in Section 5 to demonstrate implementation as well as efficiency of the proposed heuristic. Finally, conclusions and future studies appear in Section 6.

2. Problem statement

The first two or three years of the life of some SMEs is spent in incubators. During this preliminary, and yet vital, period their basic needs should be recognized and supplied by the incubator. In fact, identification and supply of these basic needs is one of the main responsibilities of incubators. The basic requirements of SMEs can be divided into two main parts: hard needs and soft needs. Basic hard needs of SMEs are mostly associated with preparation of different hardware facilities, whereas their basic soft needs consist of the required knowledge to survive and to succeed in the business environment. One of the main duties of incubators is preparation of such knowledge for SMEs.

2.1. The training programs

Training programs are the most common tools in incubators to provide the required knowledge for SMEs. They are effective to enhance personal and organizational capabilities of SMEs in different facets. Indeed, a considerable portion of the required knowledge is provided by training programs. They can also be accompanied by other educational tools, as is common, to supply soft needs. In this paper, we consider the following programs which are usually implemented in incubators (Saidi-mehrabad et al., 2008): 1) managerial principles, 2) business plan, 3) quality management, 4) marketing, 5) product development, 6) capital acquisition, 7) project control and management, and 8) financial management. One can combine some of the above mentioned programs or add some other programs. Here, we do not discuss further about the training programs for two reasons. First, a lot of research has been devoted to determine the appropriate training programs. Second, the proposed heuristic in this paper is independent of the number and the nature of the training programs; so, it can also be used in incubators with different training programs.

2.2. The effective criteria

The key point is that each training program should be implemented at the most suitable time. To this end, it is a good idea to consider the desired value of the criteria based on which performance of an SME is evaluated. According to the experts' opinions (Saidi-mehrabad et al., 2008), the most important criteria to evaluate SMEs are: 1) capabilities of managers, 2) strategic plan, 3) process development, 4) product development, 5) information and advertisement, 6) plan for capital acquisition, 7) scheduling and project control, 8) team working, and 9) costs management. Similarly, because of the two aforementioned reasons about the training programs, we avoid further discussion about the effective criteria to evaluate SMEs. The above criteria have dynamic weights during the life period of SMEs in incubators. For instance, *Information and advertisement* as a criterion is not important in the early months, but it is much more important in the last months. So, a weight function depending on the time, t, can be assigned to each criterion. In addition, there is a correlation between each criterion and each training program. Some of the criteria have a very close relation with a specific training program. However, generally, there is not a one-to-one map between the criteria and the training programs. Indeed, each training program p can improve each criterion c to some extent, and the proposed heuristic is based on this sound idea. For example, the most effective training program to enhance criterion *information and advertisement* is *marketing*; so, the training program *marketing* should be implemented before the last months regarding the desired value of *information and advertisement*.

2.3. The optimization model

Here, we briefly illustrate the construction of the optimization model to schedule training programs by minimizing the sum of the weighted undesirable gaps between the obtained value and the desired value of each criterion. The main decision variables are the start time of each training program which can be determined through an optimization model. In order to construct our model we consider two facets of the problem. On the one hand, each SME is expected to fulfill each criterion to some extent, and the desired value of each criterion depends on the time. On the other hand, the obtained values of these criteria increase as the training programs are implemented. So, we can calculate the gap between the desired value of each criterion and its obtained value. Clearly, the start time of each training program directly influences these gaps. We define the objective function as the sum of these weighted undesirable gaps. The gap is undesirable whenever the desired value is greater than the obtained value. The optimization problem should find the best start time of each training program such that the undesirable gaps are minimized. At first glance, it seems that we can implement all training programs from the first month; so, we can increase the obtained values of all criteria and consequently decrease the undesirable gaps. However, we should notice that in reality both incubators and SMEs have time constraints. This implies that they cannot simultaneously implement all training programs. In other words, neither it is possible for an incubator to simultaneously implement eight training programs nor is it possible for an SME to attend simultaneously eight training programs. We should bear in mind that the main concern of SMEs is to establish and to advance their businesses rather than to participate in the training programs. They have to spend much time on their own business. Similarly, incubators are environments to help SMEs to succeed in the business rather than a college for education. Therefore, in the optimization model, we assume that at most two training programs can simultaneously be implemented. Throughout this paper we use index c for criteria, indices p and q for training programs, and index t for time. The ranges of these indices are:

$c = 1,2,...,C$

$p, q = 1,2,...,P$

$t = 1,2,...,T$ (in continuous format $0 < t \le T$).

The parameters and decision variables we need to construct the optimization model are presented in Table 1.

Table 1. Parameters and decision variables of the optimization model.

Parameters	Decision variables
$\bar{y}_c(t)$: the desired (target) value of criterion *c* at time *t*. t_c^0: the first instant from which criterion *c* has a positive desired value. $t_c^{\max}$: the first instant at which criterion *c* meets its maximum desired value. $0 \le a_{pc} \le 1$: the percentage of fulfillment of criterion *c* after training program *p* is completely implemented. $w_c(t)$: the weight of criterion *c* at time *t*. d_p: the duration of training program *p* (month).	x_p: the start time of implementation of training program *p*. $y_c(t)$: the obtained value of criterion *c* by time *t*. $y_{pc}(t)$: the contribution of training program *p* to criterion *c* by time *t*. $g_c(t)$: the undesirable gap of criterion *c* at time *t*.

The values of parameters t_c^0, $t_c^{\max}$, a_{pc}, $w_c(t)$, and d_p are determined based on the experts' opinions, and for each criterion *c*, $\bar{y}_c(t)$ is considered as a non-decreasing function of time *t*. The main decision variables are x_p's. The maximum value for each criterion is considered equal to 1, i.e.,

$$\max\{\bar{y}_c(t) \quad 0 < t \le T\} = \bar{y}_c(t_c^{\max}) = 1 \tag{1}$$

For the sake of simplicity, as indicated in Fig. 1, a linear function is considered to show the growth of the desired value for each criterion, i.e.,

$$\bar{y}_c(t) = \frac{t - t_c^0}{t_c^{\max} - t_c^0} \tag{2}$$

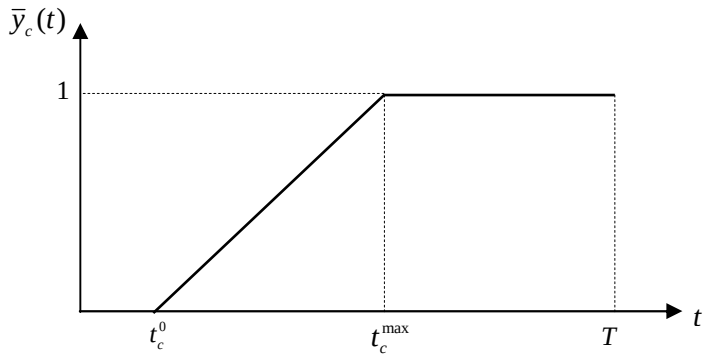

Fig. 1. The desired value of criterion *c* at time *t*.

Another assumption is that each training program *p* improves criterion *c* independent of other training programs, and the equation $\sum_{p=1}^{P} a_{pc} = 1$ does not need to hold. Now, we describe the construction of the

optimization model. Suppose that in the time interval $[x_p, x_p + d_p]$ only training program *p* is implemented; so we have,

$$y_c(x_p + d_p) = y_c(x_p) + a_{pc} \tag{3}$$

The gaps for which $\bar{y}_c(t) > y_c(t)$ are undesirable because it means that the obtained value of criterion *c* is less than its desired value. Fig. 2 shows the undesirable gaps of criterion *c*.

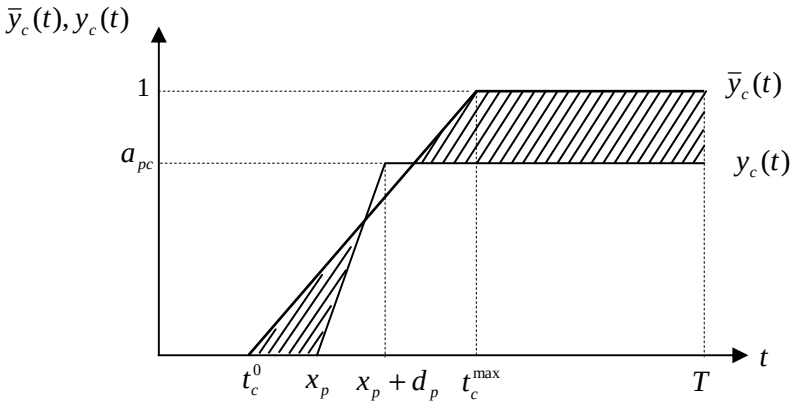

Fig. 2. The undesirable gaps for criterion *c*.

Mathematically, by considering the training programs implemented until time *t*, the undesirable gap between $\bar{y}_c(t)$ and $y_c(t)$ can be defined as,

$$g_c(t) = \max\{0, \bar{y}_c(t) - y_c(t)\} \tag{4}$$

where,

$$y_c(t) = \sum_{p=1}^{P} y_{pc}(t) \tag{5}$$

and

$$y_{pc}(t) = \begin{cases} 0 & t < x_p \\ a_{pc}\left(\dfrac{t - x_p}{d_p}\right) & x_p \le t < x_p + d_p \\ a_{pc} & t \ge x_p + d_p \end{cases} \tag{6}$$

Eq. (5) is valid due to the assumption that each training program *p* improves criterion *c* independent of other training programs. Eq. (6) shows that the contribution of training program *p* to improve criterion *c* starts from time x_p, and grows linearly until time $x_p + d_p$, and after that it remains at the level of its maximum possible contribution a_{pc}. The objective function in the optimization model is the minimization of these weighted undesirable gaps, i.e.,

$$\min \int_0^T \sum_{c=1}^{C} w_c(t) g_c(t)\, dt \tag{7}$$

The weight functions are determined in such a way that,

$$\sum_{c=1}^{C} w_c(t) = 1 \quad 0 < t \le T \tag{8}$$

To model the real constraint that at most two training programs can be implemented at each time t we define binary variables $u_p(t) \in \{0,1\}$. By defining the constraints,

$$u_p(t) = \begin{cases} 1 & \text{if } x_p \le t < x_p + d_p \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

$$\sum_{p=1}^{P} u_p(t) \le 2 \tag{10}$$

we guarantee that this assumption holds. If we neglect the assumption of simultaneously implementing at most two programs then all training programs will start from the beginning of the first month in order to minimize the undesirable gaps. Note that if program p is being implemented at time t then we must have $x_p \le t < x_p + d_p$ which consequently means $u_p(t) = 1$. The optimization problem can be summarized as follows:

$$\min \int_0^T \sum_{c=1}^{C} w_c(t) g_c(t)\, dt$$

s.t.

$$\bar{y}_c(t) = \frac{t - t_c^0}{t_c^{\max} - t_c^0} \quad c = 1,2,...,C \quad 0 < t \le T$$

$$y_{pc}(t) = \begin{cases} 0 & t < x_p \\ a_{pc}\left(\frac{t - x_p}{d_p}\right) & x_p \le t < x_p + d_p \\ a_{pc} & t \ge x_p + d_p \end{cases} \quad c = 1,2,...,C \quad p = 1,2,...,P \quad 0 < t \le T$$

$$y_c(t) = \sum_{p=1}^{P} y_{pc}(t) \quad c = 1,2,...,C \quad 0 < t \le T$$

$$g_c(t) = \max\{0, \bar{y}_c(t) - y_c(t)\} \quad c = 1,2,...,C \quad 0 < t \le T$$

$$u_p(t) = \begin{cases} 1 & \text{if } x_p \le t < x_p + d_p \\ 0 & \text{otherwise} \end{cases} \quad p = 1,2,...,P \quad 0 < t \le T$$

$$\sum_{p=1}^{P} u_p(t) \le 2 \quad 0 < t \le T$$

$$u_p(t) \in \{0,1\} \quad p = 1,2,...,P \quad 0 < t \le T$$

3. The proposed heuristic algorithm

There are two important weaknesses in the optimization model formulated in Section 2. First, the parameter *t*, time, is a continuous parameter. We might be able to solve the existing integral in the objective function, but there are many constraints which must hold for all continuous values of *t* and this is not a trivial, if not impossible, task from the mathematical point of view. In order to resolve this problem we discretize the parameter *t*. The second weakness is attributed to the existence of some conditional constraints. In order to transform conditional constraints (4), (6), and (9) into unconditional constraints we have to use many binary variables, that consequently increase the complexity of the problem from the computational point of view. Indeed, even by discretization of time, the formulated optimization model is a nonlinear mixed integer programming problem. It implies that this problem falls in one of the most difficult categories of optimization problems. Actually, the formulated problem is NP-Complete; we give a formal proof of its NP-Completeness in Section 3.2. To resolve the second weakness a heuristic algorithm is proposed. Indeed, NP-Completeness of the formulated problem justifies the necessity of developing a heuristic algorithm.

3.1. Discretization of time

The undesirable gaps can be estimated by discretizing *t* and considering time intervals as depicted in Fig. 3.

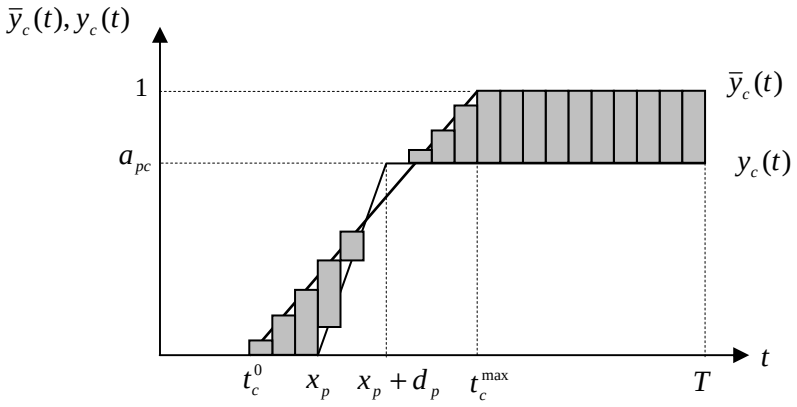

Fig. 3. The estimation of the undesirable gaps for criterion *c* by discrete intervals.

Therefore, the objective function can be rewritten as,

$$\min \sum_{c=1}^{C}\sum_{t=1}^{T} w_c(t) g_c(t) \tag{11}$$

which indicates an estimation of the weighted undesirable gaps.

Similarly, instead of constructing continuous functions, $w_c(t)$'s can be determined by using a pairwise comparison matrix between criteria at each discrete time *t*, in that they are required only in discrete times $t = 1,2,...,T$. So, at each discrete time *t* we can calculate a vector $W(t) = (w_1(t), w_2(t),..., w_C(t))$. According to Fig. 3, the desired value of criterion *c* at time *t* is estimated as,

$$\bar{y}_c(t) = \begin{cases} 0 & t = 1,2,...,t_c^0 - 1 \\ \dfrac{t + 0.5 - t_c^0}{t_c^{\max} - t_c^0} & t = t_c^0,...,t_c^{\max} - 1 \\ 1 & t = t_c^{\max},...,T \end{cases} \quad (12)$$

The modifier 0.5 in Eq. (12) implies that after discretization of $\bar{y}_c(t)$ we consider the height of the middle point of each interval as the height of the corresponding interval (see Fig. 3). Similarly, given that program *p* starts from month x_p, we can estimate $y_{pc}(t)$'s as,

$$(y_{pc}(t) \mid x_p) = \begin{cases} 0 & t = 1,2,...,x_p - 1 \\ a_{pc}\left(\dfrac{t + 0.5 - x_p}{d_p}\right) & t = x_p,...,x_p + d_p - 1 \\ a_{pc} & t = x_p + d_p,...,T \end{cases} \quad (13)$$

3.2. The NP-Completeness of the problem

The theory of NP-Completeness is only applied to decision problems. In a decision problem, we first state some problem by describing a generic instance of it. Then, a question is posed for which there are only two possible answers: *yes* or *no*. If a problem is NP-Complete then it is widely believed that the required time to answer the question by any existing algorithm increases faster than a polynomial function of the size of the instance.

The most common way to prove that a problem is NP-Complete is to consider some already known NP-Complete problem and to show that it can be transformed in polynomial time to our problem (Garey and Johnson, 1979).

Theorem:

The problem stated in Section 2.3 with discrete time is NP-Complete.

Proof:

We consider the multiprocessor scheduling problem as the already known NP-Complete problem (Garey and Johnson, 1979) which is defined as follows.

Generic instance: A set P of jobs, two parallel machines, duration time $d_p \in Z^+$ *for each* $p \in P$, *and a deadline* $K \in Z^+$.

Question: Is there a schedule for P that meets the overall deadline K?

We show that any generic instance of the multiprocessor scheduling problem can be transformed to an equivalent specific instance of our problem such that the answer to our decision problem is *yes (no)* if and only if the answer to the multiprocessor scheduling decision problem is *yes (no)*. We define the specific instance of our decision problem as follows.

Specific instance: The problem stated in Section 2.3 with discrete time, $d_p \in Z^+$ *for each* $p \in P$, $t_c^0 = K + 1 - \varepsilon$ *for each* $c \in C$ *where* ε *is a sufficiently small number,* $t_c^{\max} = K + 1$ *for each* $c \in C$, $a_{pc} = \frac{1}{P}$ *for each* $p \in P, c \in C$, *and arbitrary values for* w_c*'s. We denote the total weighted undesirable gaps by* G_{Total}.

Question: Is there a schedule for training programs with $G_{Total} = 0$*?*

If the answer to our decision problem is *yes* then all programs must have been completely implemented by the end of month *K*. This conclusion is based on the characteristics of our specific instance about a_{pc}. Indeed, in month *K*+1 we need the highest desired value, which is 1, for all criteria. If any program is not completed by the end of month *K* then we absolutely face an undesirable gap for all criteria because:

1) all programs have positive contributions on all criteria, $a_{pc} > 0$, $\forall p \in P, c \in C$,

2) the desired value for all criteria in month *K*+1 is 1,

3) the obtained value for each criterion is 1 if and only if all programs have been completely implemented (this is due to the fact that $\sum_{p=1}^{P} a_{pc} = 1$ for each $c \in C$ and it does not exceed 1).

Therefore, if the answer to our decision problem is *yes* it implies that all programs have been completely implemented by the end of month *K*. Since at most two programs are implemented at each time and the duration time of each program in our problem is the same as the duration time of its corresponding job in the multiprocessor scheduling problem we conclude that the jobs in the multiprocessor scheduling problem can be processed by the deadline *K*, i.e. the answer to the multiprocessor scheduling problem is *yes*, too.

By choosing a sufficiently small value for ε the desired value for each criterion in the months 1, 2, …, *K*-1 is zero, in month *K* is almost zero, and in month *K*+1 is 1. So, regardless of the order of programs, if all programs are implemented by the end of month *K* then $G_{Total} = 0$. It can be deduced from the last statement that if the answer to our decision problem is *no* then it is not possible to implement all programs by the end of month *K*. This implies that the jobs in the multiprocessor scheduling problem cannot be processed by the deadline *K*; hence, the answer to the multiprocessor scheduling problem is *no*, too.

□

3.3. The heuristic algorithm

Our heuristic algorithm is a greedy algorithm, i.e. at each stage it chooses the locally optimal decision. The locally optimal decision at each stage is the program *p* whose implementation yields the minimum weighted gaps from the current stage till month *T*. Each stage is the first possible month to implement a new program. After determining the optimal program at each stage we modify the desired values of all criteria. We continue till all programs are scheduled. We start from the first month and determine the most appropriate program. Since at each discrete time *t* at most two programs are allowed to be implemented we can calculate the future weighted gaps incurred by implementation of each twin combination of the programs. So, if programs *p* and *q* are selected to be implemented from the first month then the future weighted gaps are estimated as,

$$(G_{p,q} \mid x_p = 1, x_q = 1) = \sum_{c=1}^{C} \sum_{t=1}^{T} w_c(t).(g_c(t) \mid x_p = 1, x_q = 1) \tag{14}$$

where

$$(g_c(t) \mid x_p = 1, x_q = 1) = \max\{0, \bar{y}_c(t) - (y_c(t) \mid x_p = 1, x_q = 1)\} \tag{15}$$

$$(y_c(t) \mid x_p = 1, x_q = 1) = (y_{pc}(t) \mid x_p = 1) + (y_{pc}(t) \mid x_q = 1) \tag{16}$$

The last equality is based upon the assumption that each program *p* improves each criterion *c* independent of other programs. Logically, the pair (*p*,*q*) with the minimum $G_{p,q}$ should be selected for implementation from the first month. Then, the desired values for each criterion and at each month should be modified by subtracting the $g_c(t)$ from the initial desired values $\bar{y}_c(t)$.

$$\bar{y}_c^{(2)}(t) = \max\{0, \bar{y}_c(t) - (g_c(t) \mid x_p, x_q)\} \tag{17}$$

Now, in the second iteration, the earliest possible month to start the next program is,

$$t^{(2)} = \min\{1 + d_p, 1 + d_q\} \tag{18}$$

where *p* and *q* are the programs started from the first month. Among the unscheduled programs we determine a program the implementation of which from month $t^{(2)}$ incurs the minimum future weighted gaps.

$$(G_p \mid x_p = t^{(2)}) = \sum_{c=1}^{C} \sum_{t=1}^{T} w_c(t).(g_c(t) \mid x_p = t^{(2)}) \tag{19}$$

where

$$(g_c(t) \mid x_p = t^{(2)}) = \max\{0, \bar{y}_c^{(2)}(t) - (y_c(t) \mid x_p = t^{(2)})\} \tag{20}$$

$$(y_c(t) \mid x_p = t^{(2)}) = (y_{pc}(t) \mid x_p = t^{(2)}) \tag{21}$$

We continue this procedure until all training programs are scheduled. The proposed heuristic algorithm is as follows.

Begin:

Step 0 (initialization):

$r = 1$: iteration

$U = \{1,2,...,P\}$: the set of unscheduled programs

$L = \phi$: the set of last scheduled programs

$S = \phi$: the set of all scheduled programs

$$t^{(1)} = 1 \tag{22}$$

$$\bar{y}_c^{(1)}(t) = \bar{y}_c(t) \quad c = 1,2,...,C \quad t = 1,2,...,T \tag{23}$$

Step 1 (schedule two programs):

$$(y_{pc}(t) \mid x_p = t^{(r)}) = \begin{cases} 0 & t = 1,2,...,t^{(r)} - 1 \\ a_{pc}(\dfrac{t + 0.5 - t^{(r)}}{d_p}) & t = t^{(r)},...,t^{(r)} + d_p - 1 \\ a_{pc} & t = t^{(r)} + d_p,...,T \end{cases} \tag{24}$$

$$c = 1,2,...,C \quad \forall p \in U \quad t = t^{(r)},...,T$$

$$(y_c(t) \mid x_p = t^{(r)}, x_q = t^{(r)}) = (y_{pc}(t) \mid x_p = t^{(r)}) + (y_{pc}(t) \mid x_q = t^{(r)}) \tag{25}$$

$$c = 1,2,...,C \quad \forall p,q \in U \quad p \neq q \quad t = t^{(r)},...,T$$

$$(g_c(t) \mid x_p = t^{(r)}, x_q = t^{(r)}) = \max\{0, \bar{y}_c^{(r)}(t) - (y_c(t) \mid x_p = t^{(r)}, x_q = t^{(r)})\} \tag{26}$$

$$c = 1,2,...,C \quad \forall p,q \in U \quad p \neq q \quad t = t^{(r)},...,T$$

$$(G_{p,q} \mid x_p = t^{(r)}, x_q = t^{(r)}) = \sum_{c=1}^{C} \sum_{t=t^{(r)}}^{T} w_c(t).(g_c(t) \mid x_p = t^{(r)}, x_q = t^{(r)}) \quad \forall p,q \in U \quad p \neq q \tag{27}$$

$$L = \arg(\min\{(G_{p,q} \mid x_p = t^{(r)}, x_q = t^{(r)}) \quad \forall p,q \in U \quad p \neq q\}) \tag{28}$$

L contains the two indices of the unscheduled programs which incur the minimum future weighted gap.

$$g_c^{(r)}(t) = (g_c(t) \mid x_p = t^{(r)}, x_q = t^{(r)}) \quad p,q \in L \quad c = 1,2,...,C \quad t = t^{(r)},...,T \tag{29}$$

$$x_p = x_q = t^{(r)} \quad p,q \in L \tag{30}$$

Go to Step 3.

Step 2 (schedule one program):

$$(y_{pc}(t)\,|\,x_p = t^{(r)}) = \begin{cases} 0 & t = 1,2,...,t^{(r)}-1 \\ a_{pc}(\dfrac{t+0.5-t^{(r)}}{d_p}) & t = t^{(r)},...,t^{(r)}+d_p-1 \\ a_{pc} & t = t^{(r)}+d_p,...,T \end{cases} \quad (31)$$

$$c = 1,2,...,C \quad \forall p \in U \quad t = t^{(r)},...,T$$

$$(y_c(t)\,|\,x_p = t^{(r)}) = (y_{pc}(t)\,|\,x_p = t^{(r)}) \quad (32)$$

$$c = 1,2,...,C \quad \forall p \in U \quad t = t^{(r)},...,T$$

$$(g_c(t)\,|\,x_p = t^{(r)}) = \max\{0, \bar{y}_c^{(r)}(t) - (y_c(t)\,|\,x_p = t^{(r)})\} \quad (33)$$

$$c = 1,2,...,C \quad \forall p \in U \quad t = t^{(r)},...,T$$

$$(G_p\,|\,x_p = t^{(r)}) = \sum_{c=1}^{C}\sum_{t=t^{(r)}}^{T} w_c(t).(g_c(t)\,|\,x_p = t^{(r)}) \quad \forall p \in U \quad (34)$$

$$L = \arg(\min\{(G_p\,|\,x_p = t^{(r)}) \quad \forall p \in U\}) \quad (35)$$

L contains the index of an unscheduled program which incurs the minimum future weighted gap.

$$g_c^{(r)}(t) = (g_c(t)\,|\,x_p = t^{(r)}) \quad p \in L \quad c = 1,2,...,C \quad t = t^{(r)},...,T \quad (36)$$

$$x_p = t^{(r)} \quad p \in L \quad (37)$$

Go to Step 3.

Step 3 (check the termination condition):

$r = r+1$, $U = U - L$, $S = S \cup L$

If $|U| = 0$ then stop; otherwise, go to Step 4.

$|U|$ shows the cardinality of the set U.

Step 4 (update parameters):

$$N(t) = |\{p \in S \,|\, x_p \le t \text{ and } x_p + d_p \ge t\}| \quad t = 1,2,...,T \quad (38)$$

$|\{.\}|$ shows the cardinality of the set $\{.\}$. So, $N(t)$ indicates the number of programs which are being implemented in month t.

$$t^{(r)} = \min\{t = 1,2,...,T \,|\, N(t) \le 1\} \quad (39)$$

$$\bar{y}_c^{(r)}(t) = \max\{0, \bar{y}_c^{(r-1)}(t) - g_c^{(r-1)}(t)\} \quad c = 1,2,...,C \quad t = t^{(r)},...,T \quad (40)$$

If $|U| \ge 2$ and $N(t^{(r)}) = 0$ then go to Step 1; otherwise, go to Step 2.

End.

4. An equivalent job scheduling problem

Job scheduling is one of the most active research fields in operations management. In general, job scheduling means to assign jobs to machines in order to complete jobs under the existing constraints.

Many job scheduling problems are NP-Complete, and usually heuristics and meta-heuristics are developed to solve this problem. Like the course scheduling problem, many variants of the job scheduling problem have been investigated in the literature (Blazewicz et al., 1996). In some manufacturing systems we might face job scheduling problems that look like the problem stated in this paper. Consider a workshop where *C* types of products are constructed (in our problem *C* is the number of criteria). Each product consists of at most *P* different part types (in our problem *P* is the number of programs). In other words, exactly *P* different part types are manufactured in the workshop and each product consists of all or some of these parts. Suppose that there are 2 parallel machines (in our problem at most 2 programs can be implemented simultaneously), and at any time only one part type can be manufactured on each machine. Each part type might need some setup time but we can easily add its setup time to its processing time. The processing time of each part type *p* is equivalent to the duration time of implementing each program *p* in our problem, d_p. Unlike many job scheduling problems, the objective function might be associated with schedule length or mean flow time or due date of the *products* (in our problem criteria) rather than schedule length or mean flow time or due date of the *parts* (in our problem programs). Moreover, each product has an importance w_c (in our problem the weight of each criterion) which can be independent of time. For example, there might exist a due date to complete each product. Each part type *p* completes each product *c* to some extent a_{pc}. If product *c* is completed before its due date then we face inventory costs, whereas if it is completed after its due date then we face shortage costs. So, both kinds of gaps are undesirable, and we are going to minimize the total inventory as well as shortage costs of all products. By using the formulated optimization problem and the proposed heuristic algorithm we can solve such scheduling problems in manufacturing systems.

5. A real-life numerical example

This section provides a real-life numerical example to show the implementation of the proposed heuristic algorithm. As mentioned before, the parameters of the optimization model are t_c^0, $t_c^{\max}$, a_{pc}, $w_c(t)$, and d_p. Tables 2-5 show the values of these parameters obtained based on the experts' opinions from 15 incubators.

Table 2. The percentage of fulfillment of criterion *c* after implementation of training program *p*, a_{pc}.

p/c	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
1	0.3	0.2	0.1	0.0	0.0	0.1	0.1	0.2	0.0
2	0.2	0.7	0.2	0.3	0.0	0.3	0.2	0.4	0.1
3	0.1	0.0	0.7	0.0	0.0	0.1	0.1	0.0	0.0
4	0.1	0.0	0.0	0.1	0.5	0.0	0.0	0.0	0.1
5	0.1	0.2	0.0	0.7	0.3	0.0	0.0	0.0	0.0
6	0.1	0.0	0.0	0.0	0.0	0.6	0.0	0.0	0.2
7	0.1	0.2	0.0	0.0	0.0	0.0	0.6	0.2	0.1
8	0.1	0.0	0.0	0.0	0.0	0.3	0.0	0.0	0.4

Table 3. The first point with a positive desired value and the first point with the maximum desired value for each criterion.

c	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
t_c^0	0	0	6	3	9	0	0	0	4
$t_c^{\max}$	3	4	8	15	14	8	6	8	18

Table 4. The weight of criterion *c* at time *t*, $w_c(t)$.

t/c	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
1	1.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
2	0.47	0.24	0.00	0.00	0.00	0.09	0.10	0.07	0.03
3	0.31	0.31	0.00	0.00	0.00	0.12	0.12	0.10	0.04
4	0.24	0.26	0.00	0.04	0.00	0.15	0.15	0.11	0.05
5	0.20	0.22	0.00	0.05	0.00	0.17	0.17	0.13	0.06
6	0.16	0.19	0.00	0.06	0.00	0.19	0.19	0.14	0.07
7	0.14	0.15	0.06	0.07	0.03	0.17	0.17	0.14	0.07
8	0.11	0.13	0.10	0.08	0.05	0.15	0.15	0.15	0.08
9	0.10	0.10	0.14	0.09	0.07	0.14	0.14	0.14	0.08
10	0.09	0.09	0.13	0.10	0.09	0.13	0.14	0.14	0.09
11	0.08	0.08	0.13	0.12	0.11	0.12	0.13	0.13	0.10
12	0.07	0.07	0.13	0.13	0.13	0.10	0.13	0.13	0.11
13	0.07	0.06	0.13	0.13	0.13	0.10	0.12	0.14	0.12
14	0.06	0.05	0.14	0.14	0.14	0.09	0.11	0.14	0.13
15	0.06	0.04	0.14	0.14	0.14	0.09	0.11	0.14	0.14
16	0.06	0.03	0.13	0.15	0.15	0.08	0.10	0.15	0.15
17	0.05	0.01	0.12	0.16	0.16	0.08	0.10	0.16	0.16
18	0.05	0.00	0.11	0.17	0.17	0.07	0.09	0.17	0.17
19	0.05	0.00	0.11	0.17	0.16	0.07	0.08	0.16	0.20
20	0.04	0.00	0.10	0.16	0.16	0.06	0.08	0.16	0.24
21	0.04	0.00	0.10	0.15	0.15	0.06	0.07	0.15	0.28
22	0.04	0.00	0.08	0.13	0.13	0.05	0.06	0.13	0.38
23	0.03	0.00	0.06	0.09	0.09	0.04	0.05	0.09	0.55
24	0.01	0.00	0.02	0.04	0.04	0.02	0.02	0.05	0.80

Table 5. Duration of training program *p* (month).

p	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**
d_p	5	4	3	2	3	3	5	5

Before starting the heuristic algorithm, we can easily calculate the desired value of each criterion *c* at each time *t*, using Eq. (12). In the first iteration we obtain the incurred weighted undesirable gaps for each pair of programs as indicated in the upper part of Table 6. In this table, letter *s* refers to the scheduled programs.

Table 6. The incurred weighted sum of weighted undesirable gaps in each iteration.

Iter.	Gap	*p/q*	1	2	3	4	5	6	7	8
1	$G_{p,q}^{(1)}$	**1**	-----	13.65	17.05	17.25	16.39	16.75	16.18	16.72
		2	-----	-----	13.75	13.97	13.30	13.57	12.88	13.45
		3	-----	-----	-----	17.34	16.48	16.85	16.27	16.81
		4	-----	-----	-----	-----	16.82	17.04	16.48	17.05
		5	-----	-----	-----	-----	-----	16.16	15.61	16.15
		6	-----	-----	-----	-----	-----	-----	15.98	16.69
		7	-----	-----	-----	-----	-----	-----	-----	15.99
		8	-----	-----	-----	-----	-----	-----	-----	-----
2	$G_q^{(2)}$		9.80	s	9.39	9.53	9.10	9.32	s	9.19
3	$G_q^{(3)}$		7.60	s	7.07	7.44	s	7.04	s	6.85
4	$G_q^{(4)}$		5.28	s	4.76	5.00	s	5.05	s	s
5	$G_{p,q}^{(5)}$	**1**	-----	s	s	2.01	s	2.36	s	s
		4	-----	s	s	-----	s	1.81	s	s
		6	-----	s	s	-----	s	-----	s	s
6	$G_q^{(6)}$		1.07	s	s	s	s	s	s	s

Since $G_{2,7}^{(1)} = 12.88$ is the minimum value in iteration 1 we choose programs 2 and 7 as the first two programs which should be implemented. So,

$L = \{2,7\}$, $U = \{1,3,4,5,6,8\}$, $S = \{2,7\}$, $x_2 = x_7 = 1$, $t^{(2)} = 5$.

The new desired values for each criterion are calculated in step 4 of the algorithm. Then, the new incurred gaps from month $t^{(2)} = 5$ on are calculated for each program in the set of unscheduled programs *U*. These gaps for iteration 2 have been shown in its corresponding row in Table 6. Since $G_5^{(2)} = 9.10$ is the minimum value we choose program 5 as the training program which should be implemented as the third program. So,

$L = \{5\}$, $U = \{1,3,4,6,8\}$, $S = \{2,7,5\}$, $x_5 = 5$, $t^{(3)} = 6$.

The results of the next iterations have been presented in their corresponding rows in Table 6. Note that in iteration 5 we need to choose two programs because programs 2, 3, 5, 7, and 8 have been completed before month $t^{(5)} = 11$. This implies that in month 11 we can start two programs rather than only one program. By implementation of the proposed heuristic algorithm we obtain the final solution as indicated in Table 7. Moreover, Figure 4 shows a schematic representation of the final solution.

Table 7. Values of the decision variables for the numerical example.

p	1	2	3	4	5	6	7	8
x_p	13	1	8	11	5	11	1	6

Program	Month																
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**	**16**	**17**
	2	2	2	2													
	7	7	7	7	7												
					5	5	5										
						8	8	8	8	8							
								3	3	3							
											4	4					
											6	6	6				
													1	1	1	1	1

Fig. 4. Schedule of the training programs for the numerical example.

The value of the objective function (the sum of the weighted undesirable gaps) for the presented solution in Fig. 4 is 4.51. The final solution can be considered as a schedule for training programs in incubators in which special attention has been paid to the SMEs' requirements. Even though the parameters, criteria, and training programs might change, the structure of the optimization model and the proposed heuristic algorithm do not change. So, they remain valid for different training programs and criteria.

In order to compare the value of the objective function obtained by the heuristic algorithm, we address two well-known methods which are often used in incubators. The first usual simple method to schedule training programs in incubators is based upon the logic that longest training programs should be implemented as soon as possible. To this end, d_p's of the programs are sorted in non-increasing order. The programs associated with the two greatest d_p's are scheduled from the first month. Then, the program associated with the third greatest d_p is scheduled, and we continue till all programs are scheduled. By this method, the programs are scheduled in the following order: 1, 7, 8, 2, 3, 5, 6, 4. As a result, the start times of programs 1 to 8 are 1, 6, 10, 14, 11, 13, 1, 6, respectively. The sum of the weighted undesirable gaps for such scheduling is 5.63 which is 5.63-4.51=1.12 units worse than the corresponding value obtained by implementation of the heuristic algorithm.

The second method specifies the set of those criteria for which the desired value must be achieved very early (earliest criteria). Then, the training programs are sorted according to their impact to improve these criteria. In other words, we first specify the set of those criteria with the smallest value t_p^0; let us call this set *A*. Then, we sort the programs in a non-increasing order of the summation of their impacts to improve the existing programs in *A*, i.e. in non-increasing order of $\sum_{c\in A} a_{pc}$. So, the two programs with the biggest values of $\sum_{c\in A} a_{pc}$ are scheduled first. By this method, the programs are

scheduled in the following order: 2, 7, 1, 6, 8, 3, 5, 4. As a result, the start times of programs 1 to 8 are 5, 1, 10, 14, 13, 6, 1, 9, respectively. The sum of the weighted undesirable gaps for such scheduling is 4.60 which is 4.60-4.51=0.09 unit worse than the corresponding value obtained by implementation of the heuristic algorithm.

We can calculate a lower bound for the objective function by letting all programs start from the first month. If we relax (neglect) the constraints in Eqs. (9)-(10) the best value of the objective function is obtained by starting all programs as soon as possible, i.e. from the first month. This is due to the fact that $y_{pc}(t)$ is a non-increasing function of x_p. This means that the greatest value of $y_{pc}(t)$ is gained when x_p takes its smallest value, i.e. 1. By having the greatest values of $y_{pc}(t)$ we obtain the greatest values of $y_c(t)$ according to Eq. (5) and consequently the smallest values for undesirable gaps $g_c(t)$ according to Eq. (4). Such assignment leads to the value 1.77 as the sum of the weighted undesirable gaps which can be considered as a lower bound. Another lower bound is obtained if we relax the integrality of integer variables in the problem formulated in Section 2. By such relaxation we obtain 3.82 for the sum of the weighted undesirable gaps, which is obviously a stronger lower bound. Table 8 shows the weighted sum of the undesirable gaps for the heuristic algorithm, two other mentioned methods, and the lower bound.

Table 8. The weighted sum of undesirable gaps for different methods.

Heuristic algorithm	Scheduling length	Earliest criteria	Lower bound (LP relaxation)
4.51	5.63	4.60	3.82

6. Conclusions and future studies

We have discussed the construction of an optimization model to determine the most suitable times to implement training programs in incubators. This model is based on the idea that each training program is able to enhance, to some extent, the criteria by which the performance of an SME is evaluated. The weaknesses of the optimization model about modeling and computations have been briefly discussed. In order to resolve these two weaknesses we have discretized the time and proposed a heuristic method, respectively. Since our problem is NP-Complete it is worth to develop a heuristic method to find the final solution. A real-life numerical example based on the experts' opinions has been presented to show the implementation of the proposed heuristic. The numerical results of the proposed heuristic have been compared with other methods in order to demonstrate its efficiency. These comparisons show that the proposed heuristic performs better than two other methods, and its value of the objective function is not too far from the lower bound of the objective function. As one of the future studies, duration times of the training programs can be considered as decision variables. We

might be able to consider duration time of each program as a non-increasing function of the length of the time slots on each day during which the program is implemented. In this case, the number of programs which can be simultaneously implemented will depend on the maximum hours per day that each SME can spend for the training programs. Another interesting topic for future research is adjusting the proposed optimization model and heuristic algorithm to fit with the job scheduling problem discussed at the end of Section 3.

Acknowledgement

The authors would like to heartily thank Prof. Yves Crama from the HEC Management School, University of Liège for his valuable comments. The authors would also like to thank the anonymous referees for their constructive comments to improve the quality of this paper.

References

[1] Aaboen, L., (2009). Explaining incubators using firm analogy. *Technovation*. 29(10), 657-670.

[2] Alvarez-valdes, R., Crespo, E., and Tamarit, J.M., (2002). Design and implementation of a course scheduling system using tabu search. *European Journal of Operational Research*. 137, 512-523.

[3] Azimi, Z.N., (2005). Hybrid heuristics for examination timetabling problem. *Applied Mathematics and Computation*. 163(2), 705-733.

[4] Banfield, P., Jennings, P.L., Beaver, G., (1996). Competence-based training for small firms an expensive failure? *Long Range Planning*. 29(1). 94-102.

[5] Bergek, A., Norman, C., (2008). Incubator best practice: a framework. *Technovation*. 28, 20-28.

[6] Blazewicz, J., Ecker, K.H., Pesch, E., Schmidt, G., and Weglarz, J., (1996). *Scheduling computer and manufacturing processes*. Springer-Verlag, Berlin.

[7] Chrisman, J.J., McMullen, W.E., (2004). Outsider assistance as a knowledge resource for new venture survival. *Journal of Small Business Management*. 42(3), 229-244.

[8] Daskalaki, S., Birbas, T., (2005). Efficient solutions for a university timetabling problem through integer programming. *European Journal of Operational Research*. 160(1),106-120.

[9] Dimopoulou, M., Miliotis, P., (2001). Implementation of a university course and examination timetabling system. *European Journal of Operational Research*. 130(1), 202-213.

[10] Feeser, H.R., Willard, G.E., (1990). Founding strategy and performance: a comparison of high and low growth high-tech firms. *Strategic Management Journal*. 11(2), 87-98.

[11] Garey, M.R., Johnson, D.S., (1979). *Computers and intractability: A guide to the theory of NP-Completeness*. Freeman, New York.

[12] Grimaldi, R., Grandi, A., (2005). Business incubators and new venture creation: an assessment of incubating models. *Technovation*. 25(2), 111-121.

[13] Head, C., Shaban, S., (2007). A heuristic approach to simultaneous course/student timetabling. *Computers and Operations Research*. 34(4), 919-933.

[14] Juang, Y.S., Lin, S.S., and Kao, H.P., (2007). An adaptive scheduling system with genetic algorithms for arranging employee training programs. *Expert Systems with Applications*. 33, 642-651.
[15] Lu, Z., Hao, J.K., (2010). Adaptive Tabu Search for course timetabling. *European Journal of Operational Research*. 200(1), 235-244.
[16] Martocchio, J.J., Baldwin, T.T., (1997). The evolution of strategic organizational training. *Research in Personnel and Human Resources Management*. 15, 1-46.
[17] Saidi-mehrabad, M., Rezaei Sadrabadi, M., Mohammadian, I., (2008). A new method to fuzzy modeling and its application in performance evaluation of tenants in incubators. *International Journal of Advanced Manufacturing Technology*. 37, 191-201.
[18] Salas, E., Cannon-Bowers, J.A., (2001). The science of training: a decade of progress. *Annual Review of Psychology*. 52, 471-499.
[19] Tharenou, P., Saks, A., Moore, C., (2007). A review and critique of research on training and organizational-level outcomes. *Human Resource Management Review*. 17(3), 251-273.
[20] Thompson, G.M., (2005). Using information on unconstrained student demand to improve university course schedules. *Journal of Operations Management*. 23(2), 197-208.
[21] Valouxis, C., Efthymios, H., (2003). Constraint programming approach for college timetabling. *Computers and Operations Research*. 30(2), 1555-1572.
[22] Wang, Y.Z., (2003). Using genetic algorithm methods to solve course scheduling problems. *Expert Systems with Applications*. 25, 39-50.
[23] Zahra, S.A., Covin, J.G., (1993). Business strategy, technology policy, and firm performance. *Strategic Management Journal*. 14, 451-478.
[24] Zhang, D., Liu, Y., M'Hallah, R., Leung, S.C.H., (2010). A simulated annealing with a new neighborhood structure based algorithm for high school timetabling problems. *European Journal of Operational Research*. 203(3), 550-558.