Computers & Industrial Engineering 130 (2019) 650–660

Contents lists available at ScienceDirect

Computers & Industrial Engineering

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

A two-stage stochastic programming approach for a multi-objective course timetabling problem with courses cancelation risk

Peyman Yasari, Mohammad Ranjbar*, Negin Jamili, Mohammad-Hesam Shaelaie

Department of Industrial Engineering, Faculty of Engineering, Ferdowsi University of Mashhad, Mashhad, Iran

ARTICLE INFO

Keywords:
University course timetabling problem
Two-stage stochastic programming
Risk

ABSTRACT

We study a university course timetabling problem, where the registration is implemented in two steps: pre-registration and drop/add phases. Since the ultimate timetable is finalized based on the students' decisions made on the drop/add phase, there may be some courses need to be canceled because of not reaching to the threshold value in terms of total registered students. As a result of cancelations, some undesirable changes may be imposed on the final timetable for the students and the professors as well. In this paper, we ideally arrange course timetable, while considering the courses cancelations risk and the possible changes which may happen after the drop/add period; ultimately the undesirability is minimized in the final timetable. In order to achieve that, we optimize an objective function which consists of seven different objectives combined as a weighted sum function. As regards the solution method, we develop a two-stage stochastic programming model, a heuristic approach and a two-stage separated model where the latter one model the traditional method. The performance of all developed algorithms is then analyzed using randomly generated test instances.

1. Introduction

The timetabling problem, which is recently of much interest to researchers, is a type of resource-constrained scheduling problem involving the allocation of a set of courses to a set of time slots, in such a way to optimize a set of desirable objectives. The two well-known classic problems in this field are the class-teacher problem and the university course timetabling problem (UCTP). The class–teacher model, was introduced by Gotlieb (1963) in which a set of lecturers should be assigned to a given set of classrooms and time slots. In the mentioned problem there are several classes where a class consists of a set of students who follow exactly the same program. Assuming that all lectures have the same duration, Asratian and de Werra (2002) aimed at considering a timetabling problem corresponding to some situations which occur frequently in the basic training programs of universities and schools.

The university timetabling encompasses the examination timetabling as well as the course timetabling. The examination timetabling, firstly introduced by Carter, Laporte, and Lee (1996) and is defined as: "The assigning of examinations to a limited number of available time periods in such a way that there are no conflicts or clashes." As a brief description, this problem is to allocate a set of examinations to a given set of time slots so that none of the students are timetabled for two different exams at the same time.

The UCTP is described by Carter and Laporte (1997) as follows: "A multi-dimensional assignment problem in which students, teachers (or faculty members) are assigned to courses, course sections or classes; "events" (individual meetings between students and teachers) are assigned to classrooms and time slots." Almost similar to other timetabling issues, this problem, which is NP-hard, consists of both soft and hard constraints. As regards the hard constraints, some of those applied in literature are listed below.

I. For each time slot in each room, only one group of students and one professor can attend.
II. No more than one course is allowed for each time slot in each room.
III. The number of students attending the course must be less than or equal to the capacity of the room.
IV. The room should satisfy the features required by the course (see Cacchiani, Caprara, Roberti, & Toth, 2013; Lewis, Paechter, & McCollum, 2007).
V. Where specified, a course must be scheduled in the predefined time slot (see Goh, Kendall, & Sabar, 2017; Lewis & Thompson, 2015).
VI. To satisfy the precedence requirements, courses must be scheduled to occur in the correct order (see Babaei, Karimpour, & Hadidi, 2015).

* Corresponding author.
E-mail address: m_ranjbar@um.ac.ir (M. Ranjbar).

https://doi.org/10.1016/j.cie.2019.02.050
Received 20 October 2018; Accepted 28 February 2019
Available online 01 March 2019
0360-8352/

Additionally, there are a variety of soft constraints used in different problems, where some of them are given in the following.

I. The courses should be scheduled in a way that the empty time slots of both professors and students to be minimized (see Lewis & Thompson, 2015; Van den Broek & Hurkens, 2012).
II. Students should not be scheduled to attend only one course on a day (see Goh et al., 2017; Lewis & Thompson, 2015).
III. Courses should not be scheduled in the last time slot of a day (see Henry Obit, 2010).
IV. It is preferable that a course is taught in the specific room requested by its professor (see Lewis et al., 2007).
V. On each day, a period should be vacant for students' lunch break (see Lewis et al., 2007).

Cacchiani et al. (2013) tackled the UCTPs by two general approaches; the Curriculum-based and the Post Enrollment-based University Course Timetabling. Regarding the first approach, the students are classified into "groups" based on their curriculum and courses which required to meet their curriculum needs for their next semester; in other words, each group accounts for the curriculum as well as the courses which have to be followed and taken each semester (see Cacchiani et al., 2013). The problem is then to create the schedules of lectures where conflicts between the courses are set accordingly to the specified curriculum. In the second approach, the Post Enrollment-based Course Timetabling, the scheduling is determined based on the enrollment data of each individual student, such that all students can attend the courses on which they are enrolled. There are some recent studies in this field like Abdullah and Turabieh (2012), Cambazard, Hebrard, O'Sullivan, and Papadopoulos (2012), Ceschia, Di Gaspero, and Schaerf (2012), Méndez-Díaz, Zabala, and Miranda-Bront (2016). Lewis et al. (2007) gave a description of this type of problem used for Track Two of the Second International Timetabling Competition.

Considering the solution methods, the techniques applied for solving the UCTP are presented by Babaei et al. (2015) as follows: (a) Operational Researches (OR) based approaches, (b) Metaheuristic methods, (c) multi criteria and multi objective techniques, (d) intelligent novel approaches and (e) distributed multi agent systems approach. The graph coloring method, one of the OR based techniques, was firstly addressed by Welsh and Powell (1967), however, this method failed to solve the instances with pre-assigned courses. Using an almost similar approach, De Werra (1985) proposed a formulation for problem in terms of edge coloring in bipartite multigraph where the nodes are the classes and the teachers, constraints are defined by edges and each period corresponds to a color. In order to develop a more efficient approach in terms of running time and fitness performance, Asham, Soliman, and Ramadan (2011) utilized graph coloring and Genetic Algorithms (GA) as a hybrid solution.

According to the literature, the linear programming which is a subset of operational research methods, is widely used in scheduling problems. The model developed by Bakir and Aksop (2008) is based on the mentioned approach, where an optimum course scheduling timetable is achieved by using a 0–1 integer programming model, in which both students' and lecturers' dissatisfaction is minimized and set of constraints are implemented to apply the rules.

Looking closer at the literature, we should cite Lewis (2008) as some who categorized the metaheuristic algorithms for timetabling into three groups: (a) One-Stage Optimization Algorithms, (b) Two-Stage Optimization Algorithms and (c) Algorithms that allow relaxation. Alvarez-Valdes, Crespo, and Tamarit (2002) used Tabu Search (TS) to solve the course timetabling problem in three phases and built a timetable which is suitable for the characteristics of the Spanish university system. This solution method was also applied by Goh et al. (2017), where it was combined with Simulated Annealing (SA) to improve the solution quality of feasible solutions. The combinatorial optimization is also addressed by Tuga, Berretta, and Mendes (2007), in which the authors provided the combination of Kempe neighboring chain in the SA algorithm. In their approach, a feasible solution created based on a heuristic-based graph and SA algorithm, is used to minimize the violations of soft constraints. A different metaheuristic method for solving the university timetabling problem is investigated by Khonggamnerd and Innet (2009), where a genetic algorithm model was applied for improving the effectiveness of automatic arranging university timetable. Song, Liu, Tang, Peng, and Chen (2018) focused on an iterated local search algorithm to find a feasible solution for the UCTP. The computational results in this paper shows their developed algorithm achieves highly competitive results compared with the existing algorithms.

The fuzzy methodology, considered in the group of intelligent novel approaches, was firstly introduced by Zadeh (1965) and has been widely utilized in a wide range of real-world applications. In the UCTP, Chaudhuri and De (2010) developed a fuzzy genetic algorithm so that fuzzy set models measure the violation of soft constraints in the fitness function to control the inherent uncertainty and vagueness involved in real life data.

However, in real world situations, there is an inherent uncertainty about the total number of students who are going to enroll for each course. Since the final status of each course, whether it is kept in the timetable or is canceled, is determined based on the enrollment data of all the students, it is definitely efficient to handle the ambiguity on this issue. We know that the post-enrollment course timetabling models are to tackle this problem by scheduling based on the selections made by the students, not the curricula of the university. However, in some universities, the period between student's registration and the next semester is short and the grades are still not announced in the enrollment period. Thus, there may be a significant modification in the finalized student lists after the drop/add period. It is worth mentioning that the importance of coping with this uncertainty is highlighted when the overall satisfaction of students and professors are taken into account. Minimizing the time gap, also called the idle time between two classes of a student or a professor, should be considered as a soft constraint in this problem, which has not appeared in the literature. In other words, it is more desirable for students and professors to have a compact timetable and not to experience a time gap between the two courses scheduled in a given day.

In this paper, it is assumed that for each semester an initial course timetable is constructed in which the preference of both groups of professors and students is included. The drop/add period then begins after the grades announcement. Due to the newly made modifications and the policy of university on the minimum number of students participated in a class, there may be some classes that have to get canceled because of not reaching the minimum threshold. In this phase, we make the following assumptions; at he classes in which the number of students is remarkably less than a predefined threshold will be definitely "canceled" and those with a great number of registrations are "fixed". The status of classes consisting of a number of students that is negligibly less than the threshold will temporarily change to "not-fixed" in order to be determined by the decision of the administration office; and eventually, the status of not-fixed classes will be changed into either fixed or canceled. The status of each course is probabilistic and is described based on different scenarios. We develop models to construct initial timetable and decide about not-fixed courses such that expected value of the objective function is optimized.

This paper makes the following contributions; we consider UCTP with the cancelation risk of courses for the first time. We develop a two-stage stochastic programming model that considers all possible scenarios may happen to the courses status after the initial registration phase, in order to provide a feasible timetable maximizing the expected value of satisfaction level among all teachers and students. Although the two-stage stochastic programming approach has not been used in the literature for the UCTP, it has been applied for other scheduling problems like the nurse scheduling problem in the research work of Kim

and Mehrotra (2015). A noteworthy feature of this problem is to develop a practical technique for the university course timetabling where the registration is implemented in two steps (pre-registration and drop/add phases), and the possible scenarios for the second step are regarded in the decision-making process of the final timetabling. Since the developed model may be intractable for large size instances, we develop a heuristic algorithm to solve the problem by considering all possible scenarios. Also, the traditional scheduling approach is modeled using a two-stage separated model. Finally, all developed approaches are compared using extensive computational results.

The remainder of this paper is organized as follows. Section 2 deals with modeling the problem as a linear integer programming and solution methods are sketched in Section 3. Section 4 is devoted to the computational study and evaluation of the developed algorithms. Finally, Section 5 concludes the paper while providing some future research directions.

2. Problem description and modelling

In this section, we first describe the problem and notations in Section 2.1. Next, in Sections 2.2 and 2.3 we develop a two-stage stochastic programming model as an integrated model in which the initial timetable is created and also based each scenario, the final status of each not-fixed course is determined. Also, in Section 2.4 we develop a two-stage separated model describing the traditional approach of timetabling. In this approach, the initial timetable is constructed using a model and based on the realization of a scenario, a second model is solved to decide about not-fixed courses.

2.1. Problem description

Being formulated in mathematical terms, the problem comprises a set of n courses $C = \{1, \cdots, n\}$, a set of time slots $T = \{1, \cdots, nts\}$ which includes $D = \{1, \cdots, wd\}$ as the set of days in a week in which the courses are offered, such that each day is divided into td time slots. We assume there are two types of courses. Let C_1 as the set of cu^1-credit courses requiring one session per week and C_2 as the set of cu^2-credit courses requiring two sessions per week. $P = \{1, \cdots, m\}$ denotes the set of professors, in which the set of courses offered by professor j is defined by PC_j. Each professor and student has his/her own maximum limit for the number of classes per day, determined by lp^{max} and ls^{max}, respectively. In a semester, professor j and student f are required to have the minimum of lbc_j and lbs_f course credits, respectively. The set of students F is partitioned into subsets based upon their entrance semester to the university where each F_f is called a group. A binary parameter pts_{jt} is to express the availability of professor j to offer a course in time slot t. NC denotes the number of rooms which are available at each time slot; it is worth mentioning that rooms are characterized in terms of features and size.

The pre-assignment constraints are also considered in this problem. In order to do so, three binary parameters are defined as follows. Let pa_{it} be a binary parameter that gets value 1 if course $i \in C_1$ is pre-assigned to time slot t. We also introduce pa^1_{it} equals to 1 where the first session of course $i \in C_2$ is assigned to time slot t during the week, and pa^2_{it} takes the value 1 when the second session of this course is scheduled in time slot t. It is to be noted that in order to evaluate the satisfaction level, parameter PF_{it} is defined as the value brought by assigning course i to time slot t. This parameter is determined by some questionnaires acquired from professors and students. For example, for some difficult courses, the beginning time slots of each day are much better than time slots after lunch. We also define $FTS = \{1, td + 1, 2td + 1, \cdots, (wd - 1)td + 1\}$ as the set of the first time slots of the working days per week and $LTS = \{td, 2td, \cdots, wd \times td\}$ as the set of the last time slots of the working days per week.

The first step of the proposed integrated approach is to generate a feasible course timetable in which all the n courses are initially scheduled. Followed by this phase, the second stage is designed in order to improve the constructed schedule. For this matter, the cancelation risk of courses is going to be taken into account. It should be noticed that a limited number of proposed courses have the risk of cancelation and we indicate them by the set $C_{st} \subseteq C$. Let $\Omega = \{1, \cdots, k\}$ be the set of scenarios and consider $p(\omega)$ as the probability of scenario ω, in which a subset of C_{st} called E_ω is defined as the "canceled" courses, and $A_\omega \subseteq C_{st}$ includes those courses with "fixed" status. The remained courses, which are denoted by the set $DT_\omega = C_{st} \backslash (E_\omega \cup A_\omega)$, are then introduced as "not-fixed".

2.2. The integrated model

In this section, we provide an integrated model for the addressed problem, so that all the possible scenarios of the second step and their realization probabilities can be investigated in one stage. Table 1 summarizes the decision variables required to formulate the problem, separately specified for the two stages. Tables 1 and 2 describe the decision variables of the first stage (here-and-now) as well as the second stage (wait-and-see) respectively.

In the proposed formulation, the idle times denoted by $Gapp$ and $Gaps$ in Tables 1 and 2, are defined for the time slots in the middle of a day, which take the value of 1 where no course is assigned to the given middle time slot, and at least two time slots before and after that period are occupied by some courses. The courses with not-fixed status, included in set DT_ω, are also changed to either fixed or canceled subject to the soft constraints considered in the following objective functions. Eqs. (1)–(7) are, respectively, to minimize the overlapping conflicts, maximize the overall assignment value, minimize the working days of both professors and students, minimize the professors' as well as the students' idle time and minimize the number of course credits assigned to the professors whose offered courses credits are below the minimum number of credits (lbc_j). It is to be mentioned that all the functions are separately investigated for each scenario $\omega \in \Omega$.

Table 1
The first stage variables.

X_{it}	A binary variable that takes the value of 1 if course $i \in C^1$ is assigned to time slot t and takes 0, otherwise	$Gapp_t^{j1}$	An auxiliary binary variable defined for calculating the idle time, indicating if professor j offers at least one course in any time slot before time slot$t$
X^1_{it}	A binary variable that takes the value of 1 if the first session of course $i \in C^2$ is assigned to time slot t and takes 0, otherwise	$Gapp_t^{j2}$	An auxiliary binary variable defined for calculating the idle time, indicating if professor j offers at least one course in any time slot after time slot$t$
X^2_{it}	A binary variable that takes the value of 1 if the second session of course $i \in C^2$ is assigned to time slot t and takes 0, otherwise	$Gaps_t^f$	An auxiliary binary variable defined for calculating the idle time, indicating if group f is idle in time slot t and takes 0, otherwise
Z_{fd}	A binary variable that takes the value of 1 if at least one element of F_f is scheduled in day d and takes 0, otherwise	$Gaps_t^{f1}$	An auxiliary binary variable defined for calculating the idle time, indicating if at least one course is offered for group f in any time slot before time slot$t$
W_{jd}	A binary variable that takes the value of 1 if at least one element of PC_j is scheduled in day d and takes 0, otherwise	$Gaps_t^{f2}$	An auxiliary binary variable defined for calculating the idle time, indicating if at least one course is offered for group f in any time slot after time slot$t$
$Gapp_t^j$	A binary variable that takes the value of 1 if professor j is idle in time slot t and takes 0, otherwise	VP_j	The penalty cost which represents the difference between lbc_j and the total course credits offered by professor j, in case of violation of the minimum course credits constraint
$Y_{ii'}$	A binary variable that takes the value of 1 if course i overlaps with course $i'$and takes 0, otherwise		

Table 2
The second stage variables.

$Z_{fd\omega}$	A binary variable that takes the value of 1 if in scenario ω, at least one course of F_f is scheduled in day d and takes 0, otherwise	$Gapp_{t\omega}^{j1}$	An auxiliary binary variable defined for calculating the idle time, indicating if in scenario ω professor j offers at least one course in any time slot before time slot t
$W_{jd\omega}$	A binary variable that takes the value of 1 if in scenario ω, at least one course of PC_j is scheduled in day d and takes 0, otherwise	$Gapp_{t\omega}^{j2}$	An auxiliary binary variable defined for calculating the idle time, indicating if in scenario ω professor j offers at least one course in any time slot after time slot t
$X_{it\omega}$	A binary variable that takes the value of 1 if in scenario ω, course $i \in C^1$ is assigned to time slot t and takes 0, otherwise	$Gaps_{t\omega}^{f}$	A binary variable that takes the value of 1 if in scenario ω, group f is idle in time slot t and takes 0, otherwise
$X_{it\omega}^{1}$	A binary variable that takes the value of 1 if in scenario ω, the first session of course $i \in C^2$ is assigned to time slot t and takes 0, otherwise	$Gaps_{t\omega}^{f1}$	An auxiliary binary variable defined for calculating the idle time, indicating if in scenario ω at least one course is offered for group f in any time slot before time slot t
$X_{it\omega}^{2}$	A binary variable that takes the value of 1 if in scenario ω, the second session of course $i \in C^2$ is assigned to time slot t and takes 0, otherwise	$Gaps_{t\omega}^{f2}$	An auxiliary binary variable defined for calculating the idle time, indicating if in scenario ω at least one course is offered for group f in any time slot after time slot t
$Gapp_{t\omega}^{j}$	A binary variable that takes the value of 1 if in scenario ω, professor j is idle in time slot t and takes 0, otherwise	$VP_{j\omega}$	The penalty cost which represents the difference between lbc_j and the total course credits offered by professor j in scenario ω, in case of violation of the minimum course credits constraint
$Y_{ii'\omega}$	A binary variable that takes the value of 0 if the two courses i and i' are assigned without overlapping and takes 1, otherwise		

$$f_{1\omega} = \min\left(p(\omega) \times \sum_{\forall i \neq i', i \in C, i' \in C} Y_{ii'\omega}\right) \quad (1)$$

$$f_{2\omega} = max\left(p(\omega) \times \left(\sum_{i \in C^1} \sum_{t \in TS} X_{it\omega} PF_{it} + \sum_{i \in C^2} \sum_{t \in TS} (X_{it\omega}^1 + X_{it\omega}^2) PF_{it}\right)\right) \quad (2)$$

$$f_{3\omega} = min\left(p(\omega) \times \sum_{\forall j \in P, \forall d \in D} W_{jd\omega}\right) \quad (3)$$

$$f_{4\omega} = min\left(p(\omega) \times \sum_{\forall f \in F_f, \forall d \in D} Z_{fd\omega}\right) \quad (4)$$

$$f_{5\omega} = min\left(p(\omega) \times \sum_{\forall t \in TS \setminus (FTS \cup LTS)} \sum_{j \in P} Gapp_{t\omega}^{j}\right) \quad (5)$$

$$f_{6\omega} = min\left(p(\omega) \times \sum_{\forall t \in TS \setminus (FTS \cup LTS)} \sum_{f \in F_f} Gaps_{t\omega}^{f}\right) \quad (6)$$

$$f_{7\omega} = min\left(p(\omega) \times \sum_{j \in P} VP_{j\omega}\right) \quad (7)$$

The mathematical formulation of the problem reads as follows.

$$Min \;\; O.F. = \sum_{\forall \omega \in \Omega} \left(W_1 \frac{f_{1\omega} - f_{1\omega}^-}{f_{1\omega}^+ - f_{1\omega}^-} - W_2 \frac{f_{2\omega} - f_{2\omega}^-}{f_{2\omega}^+ - f_{2\omega}^-} + W_3 \frac{f_{3\omega} - f_{3\omega}^-}{f_{3\omega}^+ - f_{3\omega}^-} + W_4 \frac{f_{4\omega} - f_{4\omega}^-}{f_{4\omega}^+ - f_{4\omega}^-} + W_5 \frac{f_{5\omega} - f_{5\omega}^-}{f_{5\omega}^+ - f_{5\omega}^-} + W_6 \frac{f_{6\omega} - f_{6\omega}^-}{f_{6\omega}^+ - f_{6\omega}^-} + W_7 \frac{f_{7\omega} - f_{7\omega}^-}{f_{7\omega}^+ - f_{7\omega}^-}\right) \quad (8)$$

subjected to:

$$\sum_{t \in TS} X_{it} = 1; \forall\, i \in C^1 \quad (9)$$

$$\sum_{t \in TS} X_{it}^1 = 1; \forall\, i \in C^2 \quad (10)$$

$$\sum_{t \in TS} X_{it}^2 = 1; \forall\, i \in C^2 \quad (11)$$

$$X_{it} \leq pts_{jt}; \forall\, i \in PC_j \cap C^1, \forall\, t \in TS, \forall\, j \in P \quad (12)$$

$$X_{it}^1 \leq pts_{jt}; \forall\, i \in PC_j \cap C^2, \forall\, t \in TS, \forall\, j \in P \quad (13)$$

$$X_{it}^2 \leq pts_{jt}; \forall\, i \in PC_j \cap C^2, \forall\, t \in TS, \forall\, j \in P \quad (14)$$

$$\sum_{t=\tau}^{\tau+2td-1} (X_{it}^1 + X_{it}^2) \leq 1; \forall\, \tau \in FTS, \forall\, i \in C^2 \quad (15)$$

$$X_{it} \geq pa_{it}; \forall\, t \in TS, \forall\, i \in C^1 \quad (16)$$

$$X_{it}^1 \geq pa_{it}^1; \forall\, t \in TS, \forall\, i \in C^2 \quad (17)$$

$$X_{it}^2 \geq pa_{it}^2; \forall\, t \in TS, \forall\, i \in C^2 \quad (18)$$

$$\sum_{\forall i \in C^1} X_{it} + \sum_{\forall i \in C^2} (X_{it}^1 + X_{it}^2) \leq NC; \forall\, t \in TS \quad (19)$$

$$\sum_{\forall i \in C^1 \cap P_j} X_{it} + \sum_{\forall i \in C^2 \cap P_j} (X_{it}^1 + X_{it}^2) \leq 1; \forall\, t \in TS, \forall\, j \in P \quad (20)$$

$$\sum_{t=\tau}^{\tau+td-1} \sum_{i \in (PC_j \cap C^1)} X_{it} + \sum_{t=\tau}^{\tau+td-1} \sum_{i \in (PC_j \cap C^2)} (X_{it}^1 + X_{it}^2) \leq lp^{max}; \forall\, \tau \in FTS, \forall\, j \in P \quad (21)$$

$$\sum_{t=\tau}^{\tau+td-1} \sum_{i \in (F_f \cap C^1)} X_{it} + \sum_{t=\tau}^{\tau+td-1} \sum_{i \in (F_f \cap C^2)} (X_{it}^1 + X_{it}^2) \leq ls^{max}; \forall\, \tau \in FTS, \forall\, F_f \in F \quad (22)$$

$$X_{it} + X_{i't} \leq 1; \forall\, i, i' \in (C^1 \cap F_f), \forall\, f \in F_f, \forall\, t \in TS \quad (23)$$

$$X_{it} + X_{i't}^1 \leq 1; \forall\, i (C^1 \cap F_f), \forall\, i' \in (C^2 \cap F_f), \forall\, f \in F_f, \forall\, t \in TS \quad (24)$$

$$X_{it} + X_{i't}^2 \leq 1; \forall\, i (C^1 \cap F_f), \forall\, i' \in (C^2 \cap F_f), \forall\, f \in F_f, \forall\, t \in TS \quad (25)$$

$$X_{it}^1 + X_{i't}^1 \leq 1; \forall\, i, i' \in (C^2 \cap F_f), \forall\, f \in F_f, \forall\, t \in TS \quad (26)$$

$$X_{it}^1 + X_{i't}^2 \leq 1; \forall\, i, i' \in (C^2 \cap F_f), \forall\, f \in F_f, \forall\, t \in TS \quad (27)$$

$$X_{it}^2 + X_{i't}^2 \leq 1; \forall\, i, i' \in (C^2 \cap F_f), \forall\, f \in F_f, \forall\, t \in TS \quad (28)$$

$$X_{it\omega} = X_{it}; \forall\, t \in TS, \forall\, \omega \in \Omega, \forall\, i \in (A_\omega \cap C^1) \quad (29)$$

$$X_{it\omega} = 0; \forall\, t \in TS, \forall\, \omega \in \Omega, \forall\, i \in (E_\omega \cap C^1) \quad (30)$$

$$X_{it\omega}^1 = X_{it}^1; \forall\, t \in TS, \forall\, \omega \in \Omega, \forall\, i \in (A_\omega \cap C^2) \quad (31)$$

$$X_{it\omega}^1 = 0; \forall\, t \in TS, \forall\, \omega \in \Omega, \forall\, i \in (E_\omega \cap C^2) \quad (32)$$

$$X_{it\omega}^2 = X_{it}^1; \forall\, t \in TS, \forall\, \omega \in \Omega, \forall\, i \in (A_\omega \cap C^2) \quad (33)$$

$$X_{it\omega}^2 = 0; \forall\, t \in TS, \forall\, \omega \in \Omega, \forall\, i \in (E_\omega \cap C^2) \quad (34)$$

$$X_{it\omega} \leq X_{it}; \forall\, t \in TS, \forall\, \omega \in \Omega, \forall\, i \in (DT_\omega \cap C^1) \quad (35)$$

$$X_{it\omega}^1 \leq X_{it}^1; \forall\, t \in TS, \forall\, \omega \in \Omega, \forall\, i \in (DT_\omega \cap C^2) \quad (36)$$

$$X^2_{it\omega} \le X^2_{it}; \forall\, t \in TS, \forall\, \omega \in \Omega, \forall\, i \in (DT_\omega \cap C^2) \tag{37}$$

$$\sum_{\forall t \in TS} X^2_{it\omega} = \sum_{\forall t \in TS} X^1_{it\omega}; \forall\, \omega \in \Omega, \forall\, i \in \left(DT_\omega \cap C^2\right) \tag{38}$$

$$Z_{fd\omega} \ge \frac{X_{it\omega} + X^1_{i't\omega} + X^2_{i't\omega}}{3}; \forall\, t \in TS, \forall\, d \in D, \forall\, d = \lceil \frac{t}{td} \rceil, \forall\, F_f \in F, \forall\, i \in (C^1 \cap F_f), \forall\, i' \in (C^2 \cap F_f), \forall\, \omega \in \Omega \tag{39}$$

$$W_{jd\omega} \ge \frac{X_{it\omega} + X^1_{i't\omega} + X^2_{i't\omega}}{3}; \forall\, t \in TS, \forall\, d \in D, \forall\, d = \lceil \frac{t}{td} \rceil, \forall\, j \in P, \forall\, i \in (C^1 \cap PC_j), \forall\, i' \in (C^2 \cap PC_j), \forall\, \omega \in \Omega \tag{40}$$

$$VP_{j\omega} \ge lbc_j - cu^1 \times \sum_{i \in (C^1 \cap P_j)} \sum_{t \in TS} X_{it\omega} - cu^2 \times \sum_{i \in (C^2 \cap P_j)} \sum_{t \in TS} X^1_{it\omega}; \forall\, j \in P, \forall\, \omega \in \Omega \tag{41}$$

$$Y_{ii'\omega} \ge X_{it\omega} + X_{i't\omega} - 1; \forall\, i \ne i' \in C^1, \forall\, t \in TS, \forall\, \omega \in \Omega \tag{42}$$

$$Y_{ii'\omega} \ge X_{it\omega} + X^1_{i't\omega} - 1; \forall\, i \in C^1, \forall\, i' \in C^2, \forall\, t \in TS, \forall\, \omega \in \Omega \tag{43}$$

$$Y_{ii'\omega} \ge X_{it\omega} + X^2_{i't\omega} - 1; \forall\, i \in C^1, \forall\, i' \in C^2, \forall\, t \in TS, \forall\, \omega \in \Omega \tag{44}$$

$$Y_{ii'\omega} \ge X^1_{it\omega} + X^2_{i't\omega} - 1; \forall\, i \ne i' \in C^2, \forall\, t \in TS, \forall\, \omega \in \Omega \tag{45}$$

$$Y_{ii'\omega} \ge X^1_{it\omega} + X^1_{i't\omega} - 1; \forall\, i \ne i' \in C^2, \forall\, t \in TS, \forall\, \omega \in \Omega \tag{46}$$

$$Y_{ii'\omega} \ge X^2_{it\omega} + X^2_{i't\omega} - 1; \forall\, i \ne i' \in C^2, \forall\, t \in TS, \forall\, \omega \in \Omega \tag{47}$$

$$\frac{\sum_{\tau=td\left[\frac{t}{td}\right]+1}^{t-1}\left(\sum_{i\in(C^1\cap F_f)} X_{i\tau\omega} + \sum_{i\in(C^2\cap F_f)} (X^1_{i\tau\omega} + X^2_{i\tau\omega})\right)}{M} - \left(\sum_{i\in(C^1\cap F_f)} X_{it\omega} + \sum_{i\in(C^2\cap F_f)} (X^1_{it\omega} + X^2_{it\omega})\right) \le Gaps^{f1}_{t\omega}; \forall\, f \in F_f, \forall\, t \in TS \setminus (FTS \cup LTS), \forall\, \omega \in \Omega \tag{48}$$

$$\frac{\sum_{\tau=t+1}^{td\left(\left[\frac{t}{td}\right]+1\right)}\left(\sum_{i\in(C^1\cap F_f)} X_{i\tau\omega} + \sum_{i\in(C^2\cap F_f)} (X^1_{i\tau\omega} + X^2_{i\tau\omega})\right)}{M} - \left(\sum_{i\in(C^1\cap F_f)} X_{it\omega} + \sum_{i\in(C^2\cap F_f)} (X^1_{it\omega} + X^2_{it\omega})\right) \le Gaps^{f2}_{t\omega}; \forall\, f \in F_f, \forall\, t \in TS \setminus (FTS \cup LTS), \forall\, \omega \in \Omega \tag{49}$$

$$Gaps^{f}_{t\omega} \ge Gaps^{f1}_{t\omega} + Gaps^{f2}_{t\omega} - 1; \forall\, f \in F_f, \forall\, t \in TS \setminus (FTS \cup LTS), \forall\, \omega \in \Omega \tag{50}$$

$$\frac{\sum_{\tau=td\left[\frac{t}{td}\right]+1}^{t-1}\left(\sum_{i\in(C^1\cap PC_j)} X_{i\tau\omega} + \sum_{i\in(C^2\cap PC_j)} (X^1_{i\tau\omega} + X^2_{i\tau\omega})\right)}{M} - \left(\sum_{i\in(C^1\cap PC_j)} X_{it\omega} + \sum_{i\in(C^2\cap PC_j)} (X^1_{it\omega} + X^2_{it\omega})\right) \le Gapp^{j1}_{t\omega}; \forall\, j \in P, \forall\, t \in TS \setminus (FTS \cup LTS), \forall\, \omega \in \Omega \tag{51}$$

$$\frac{\sum_{\tau=t+1}^{td\left(\left[\frac{t}{td}\right]+1\right)}\left(\sum_{i\in(C^1\cap PC_j)} X_{i\tau\omega} + \sum_{i\in(C^2\cap PC_j)} (X^1_{i\tau\omega} + X^2_{i\tau\omega})\right)}{M} - \left(\sum_{i\in(C^1\cap PC_j)} X_{it\omega} + \sum_{i\in(C^2\cap PC_j)} (X^1_{it\omega} + X^2_{it\omega})\right) \le Gapp^{j2}_{t\omega}; \forall\, j \in P, \forall\, t \in TS \setminus (FTS \cup LTS), \forall\, \omega \in \Omega \tag{52}$$

$$Gapp^{j}_{t\omega} \ge Gapp^{j1}_{t\omega} + Gapp^{j2}_{t\omega} - 1; \forall\, j \in P, \forall\, t \in TS (FTS \cup LTS), \forall\, \omega \in \Omega \tag{53}$$

$$cu^1 \sum_{i\in(C^1\cap F_f)} \sum_{t\in TS} X_{it\omega} + cu^2 \sum_{i\in(C^2\cap F_f)} \sum_{t\in TS} X^1_{it\omega} \ge lbs_f; \forall\, f \in F_f, \forall\, \omega \in \Omega \tag{54}$$

$$X_{it}, X^1_{it}, X^2_{it}, X_{it\omega}, X^1_{it\omega}, X^2_{it\omega}, Y_{ii'\omega} \in \{0, 1\}; \forall\, i, i' \in C, \forall\, t \in TS, \forall\, \omega \in \Omega$$

$$Gapp^{j}_{t\omega}, Gapp^{j1}_{t\omega}, Gapp^{j2}_{t\omega}, Gaps^{f}_{t\omega}, Gaps^{f1}_{t\omega}, Gaps^{f2}_{t\omega}, Z_{fd\omega}, W_{jd\omega} \in \{0, 1\}; \forall\, j \in P, \forall\, f \in F, \forall\, d \in D, \forall\, t \in TS, \forall\, \omega \in \Omega \tag{55}$$

$$VP_{j\omega} \ge 0; \forall\, j \in P, \forall\, \omega \in \Omega \tag{56}$$

Due to the multi-objective framework considered for this model, a weighted sum objective function which is normalized based on a fuzzy evaluation system is presented in Eq. (8), where f^+_{ew} and f^-_{ew} determine the upper and lower bounds of function e, respectively. How to calculate these bounds is subsequently described in the next section. As regards the hard constraints considered in the construction of the initial timetable, constraint (9) ensures that the one-session courses in set C^1 are assigned to only one time slot during a week. This constraint is also imposed on two-session courses of set C^2 by constraints (10) and (11). Constraints (12)–(14) guarantee that if a professor of a specific course is not available at a given time slot, then no lecture of the course can be scheduled at that period. Constraint (15) establishes that different sessions of a two-session course must be scheduled in non-consecutive days. Constraints (16)–(18) are designed for the pre-assigned courses and constraint (19) stipulates the number of courses scheduled in each time slot must be less than or equal to the number of available rooms. Constraint (20) prevents the model from considering solutions with more than one course assigned to the same professor in a given time slot. The maximum number of lectures assigned to the professors and student groups in a working day is defined using constraints (21) and (22), respectively. Constraints (23)–(28) avoid conflicts in student groups so that all courses of a given group must be scheduled in different time slots.

For the second stage, the constraints are arranged as follows. Constraints (29)–(37) are for initializing and integrating the decision variable dedicated to scenarios. Constraint (38) ensures that the two sessions of a course in set C^2 must get canceled or fixed, simultaneously. Constraint (39) describes the working days of professors and student groups, so that day d is defined as a working day in case that at least one class is scheduled in it. Constraints (40) and (41) aim at calculating the total number of course credits assigned to the professors which are below their minimum limit of credits (lbc_j). Constraints (42)–(47) enumerate the course conflict. The idle time of professors and students in each scenario is determined in constraint (48)–(53) where M indicates a big positive number. Constraint (54) ensures that each student group fulfills a minimum credit requirement for each semester. Finally, the two last constraints indicates the type of the decision variables.

2.3. Determination of the lower and upper bounds

As previously mentioned, in order to achieve a weighted sum objective function, lower and upper bounds are needed. It is generally the case that the upper and lower bounds for a given model are obtained by solving the

maximization and minimization problems of that model, respectively. For this purpose, we firstly solve a specific mathematical model to reach each of the bounds. Since the probability of scenarios is an important factor included in the objective functions, the maximum and minimum of this factor are regarded in for the upper and lower bound calculation, respectively. Assume ω^* as the scenario in which all the courses have not-fixed status, in other words $DT_{\omega^*} = C_{st}$. In the following, we provide the maximization models to obtain the upper bounds of the seven objective functions in the main formulation. The lower bounds are similarly achieved using the minimization version of the same models.

$$f_1^+ = \left(\max_{\forall \omega \in \Omega} p(\omega) \times \sum_{\forall i \neq i', i \in C, i' \in C} Y_{ii'\omega^*} \right)$$

subjected to:

(9), ⋯,(38), (42), ⋯(47), (54), (55)

$$f_2^+ = \left(\max_{\forall \omega \in \Omega} p(\omega) \times \left(\sum_{i \in C^1} \sum_{t \in TS} X_{it\omega^*} PF_{it} + \sum_{i \in C^2} \sum_{t \in TS} (X^1_{it\omega^*} + X^2_{it\omega^*}) PF_{it} \right) \right)$$

subjected to:

(9), ⋯,(38), (54), (55)

$$f_3^+ = \left(\max_{\forall \omega \in \Omega} p(\omega) \times \sum_{\forall j \in P, \forall d \in D} W_{jd\omega^*} \right)$$

subjected to:

(9), ⋯,(38), (40), (54), (55)

$$f_4^+ = \left(\max_{\forall \omega \in \Omega} p(\omega) \times \sum_{\forall f \in Fj, \forall d \in D} Z_{fd\omega^*} \right)$$

subjected to:

(9), ⋯,(39), (54), (55)

$$f_5^+ = \left(\max_{\forall \omega \in \Omega} p(\omega) \times \sum_{\forall t \in TS \setminus (FTS \cup LTS)} \sum_{j \in P} Gapp^j_{t\omega^*} \right)$$

subjected to:

(9), ⋯,(38), (51), ⋯,(55)

$$f_6^+ = \left(\max_{\forall \omega \in \Omega} p(\omega) \times \sum_{\forall t \in TS \setminus (FTS \cup LTS)} \sum_{j \in P} Gaps^f_{t\omega^*} \right)$$

subjected to:

(9), ⋯,(38), (48), (49), (50), (54), (55)

$$f_7^+ = \left(\max_{\forall \omega \in \Omega} p(\omega) \times \sum_{j \in P} VP_{j\omega^*} \right)$$

subjected to:

(9), ⋯,(38), (40), (41), (55), (56)

2.4. The two-stage separated model

Since the course timetabling models presented in the literature are investigated as a single-stage procedure regardless of the probable scenarios in the second stage, in this section, we aim at comparing our developed model with those traditional methods by developing two different models for the first and second stages, separately. Consider M1 and M2 as the models proposed for the first and second stages, respectively. We show this approach in which the first stage is considered independent from the second stage as M1/M2. In this approach, firstly we generate the M1 model consisting of constraints (9)–(28) as well as Eq. (8) as the single objective function in which parameter ω is omitted. Thereafter, based upon the timetable generated by M1 model and realization of a possible scenario, we apply the M2 model in order to gain the finalized timetable. It should be mentioned that this model comprises constraints (29)–(56) and Eq. (8) such that only the realized scenario is included.

As a comparison, the integrated model allows for uncertainty when attempting to reach an initial timetable while the separated model does not consider possible scenarios.

3. Solution approach

In this section, we focus on developing a solution approach for the integrated model, in order to achieve high-quality solutions in a reasonable computational time. Meanwhile, the goal is to obtain an initial solution almost similar to the one resulted by the M1 model and then applying M2 to get the final solution. It is worth mentioning that in this procedure, the initial solution is achieved by evaluating the possible scenarios while this is ignored in implementation of M1.

The process addressed in this section is a heuristic algorithm, called HA, which results in an initial solution, and the completed procedure to present the final result is denoted by HA/M2. It is also assumed that there is at least one feasible solution achieved by this algorithm.

The following notations are considered in HA algorithm (see Table 3).

Table 3
Notations of HA.

Sol	Initial solution
Sol^*	The best found solution
LC	Set of courses to be rescheduled in the local search
ATS	Set of feasible time slots for a selected course
c	The selected course for assigning to a time slot
$n_i(ATS)$	The feasible time slots for course$i$
$Index(i)$	An index used to calculate the selection probability of course i
$Indext(t)$	An index used to calculate the selection probability of time slot$t$
$prob(i)$	The selection probability of course$i$
$prob(t)$	The selection probability of time slot t
prob	Probability vector
$Obj(t)$	The objective value in case of assigning the selected course to time slot$t$

In Algorithm 1, we show the sketch of the overall structure of HA which is similar to the GRASP algorithm. Each run will stop when the time limit is reached. HA takes a UCTP instance as an input and it contains a "While" loop. In each iteration, an initial solution which is achieved by applying Algorithm 2, is improved by using the local search procedure, shown in Algorithm 3. The algorithm then performs the next iteration in case of not finding a better solution rather than the best found one, otherwise updates the best solution and continues searching.

Algorithm 1.. Pseudo-code of HA algorithm

HA (*A UCTP instance*)

1. $Sol^* \leftarrow \varnothing$
2. **While** (*time limit has not been met*)
3. $Sol \leftarrow$ ***Initial_Solution*** (*UCTP instance*)
4. $Sol \leftarrow$ **Local_Searsh**(Sol)
5. ***If*** (*Sol is better than Sol**)
6. $Sol^* \leftarrow Sol$
7. ***End If***
8. ***End While***
9. **Return** Sol^*

End HA

Algorithm 2.. Pseudo-code of the Initial_Solution

Initial_Solution **(***A UCTP instance***)**

1. ***Input****: Initialize all input parameters based on the given UCTP instance*
2. $Sol \leftarrow preassignments$
3. **While** $(C^2 \neq \emptyset)$
4. $c \leftarrow$ Course_Selection(C^2, Sol)
5. $ATS \leftarrow$ Available_Timeslots(c, Sol)
6. **If** $(ATS = \emptyset)$
7. goto 1
8. **End If**
9. Timeslot_Selection(c, ATS)
10. Update(Sol)
11. $ATS \leftarrow$ Available_Timeslots(c, Sol)
12. **If** $(ATS = \emptyset)$
13. goto 1
14. **End If**
15. Timeslot_Selection(c, ATS)
16. Update(Sol)
17. $C^2 \leftarrow C^2 \backslash c$
18. **EndWhile**
19. **While** $(C^1 \neq \emptyset)$
20. $c \leftarrow$ Course_Selection(C^1, Sol)
21. $ATS \leftarrow$ Available_Timeslots(c, Sol)
22. **If** $(ATS = \emptyset)$
23. goto 1
24. **End If**
25. Timeslot_Selection(c, ATS)
26. Update(Sol)
27. $C^1 \leftarrow C^1 \backslash c$
28. **End While**
29. **Return** *Sol*

End Initial_Solution

The pseudo-code depicted in Algorithm 2 aims at finding an initial timetable, in which the pre-assigned courses are firstly fixed. Thereafter, the two-session and one-session courses are scheduled, respectively, since assigning the two-session courses are more difficult than others. Algorithm 3 is an indication of the course selection procedure, which is designed to choose an appropriate course for assigning to the timetable. Specifying a set of suitable time slots for the selected course (Algorithm 4), we then apply Algorithm 5 in order to reach the best time slot among the available ones and update the initial timetable with the new assignment. It is to be noted that in case of finding no feasible time slot for a specific course, the algorithm stops and apply the initial solution procedure to re-start the algorithm.

Algorithm 3.. Pseudo-code of the Course_Selection

Course_Selection **(***C, Sol***)**

1. **For** $(\forall\, i \in C)$
2. $n_i(ATS) \leftarrow$ Available_Timeslot(i, Sol)
3. **End For**
4. **For**$(\forall\, i \in C)$
5. $index(i) \leftarrow \left(\max_{\forall i \in C} (n_i(ATS)) - n_i(ATS) + 1 \right)$
6. **End For**
7. $sum \leftarrow \sum_{\forall i} index(i)$
8. **For**$(\forall\, i \in C)$
9. $prob(i) \leftarrow index(i)/sum$
10. **End For**
11. **Return** Roulette_Wheel(**prob**)

End Course_Selection

The pseudo-code shown in Algorithm 3 is to select a course in order to be assigned to the course timetable. In this regard, the courses with less flexibility are more probable to be selected. In other words, the courses with smaller set of feasible time slots, regarding as less flexible ones, shown by $index(i)$ for course i. In order to assign higher selection probability to less flexible courses, we consider $prob(i) = index(i)/sum$ where $sum = \sum_{\forall i} index(i)$. Next, a course is selected based on a biased random selection approach, called the Roulette Wheel Selection (RWS) method in the literature.

Thereafter, an appropriate time slot has to be chosen for the selected course in order to be able to properly assign it to the available timetable. As a result, Algorithm 4 is firstly used, to specify the set of available time slots. Algorithm 5 is then applied, in which the objective values are estimated while considering the available time slots for the given course. After calculating the selection probabilities for the time slots, the final time slot is selected using RWS method.

Algorithm 4.. Pseudo-code of the Available_Timeslots

Available_Timeslot **(***c, Sol***)**

1. $ATS \leftarrow \emptyset$
2. **For**$(\forall\, t \in T)$
3. **If** $Sol(t)$ is feasible for c
4. $ATS \leftarrow ATS \cup t$
5. **End If**
6. **End For**
7. **Return** *ATS*

End Timeslot_Selection

Algorithm 5.. Pseudo-code of the Timeslot_Selection

Timeslot_Selection **(***c, ATS***)**

1. **For**$(\forall\, t \in ATS)$
2. $Obj(t) \leftarrow$ objective_Function(c, t)
3. **End For**
4. **For**$(\forall\, t \in ATS)$
5. $indext(t) \leftarrow \left(\max_{\forall t \in ATS} (Obj(t)) - Obj(t) + 1 \right)$
6. **End For**
7. $sum \leftarrow \sum_{\forall t \in ATS} indext(t)$
8. **For**$(\forall\, t \in ATS)$
9. $prob(t) \leftarrow indext(t)/sum$
10. **End For**
11. **Return** Roulette_Wheel(**prob**)

End Timeslot_Selection

The local search method, presented in Algorithm 6, is a procedure developed to improve the initial solution. As is given in the following algorithm, $\alpha\%$ of the courses in the initial solution are removed from the timetable at the first step and then they are assigned to the empty sets C^1 and C^2 based on their credit. Afterwards, a procedure which is almost similar to the initial solution construction is utilized, where a greedy method is applied for selecting the courses and time slots, as well. We call these procedures as *Greedy-Course Selection* and *Greedy_Timeslot_Selection* procedures.

It should be noted that the selection procedures in this algorithm may sometimes lead to an infeasible solution. Regarding the greedy methods, no specific process is designed to escape the infeasibility in case of need. As a result, when facing such a problem, the algorithm reassigns all the removed courses and continues the process. Ultimately, the algorithm either improve or just return untouched initial solution in worst-case scenario.

Algorithm 6.. Pseudo-code of the Local_Search

Local_Search **(*Sol*)**

1. *LC* ← α% *of all courses which scheduled in Sol and do not have any pre-assignment.*
2. $C^1 \leftarrow \varnothing$ *and* $C^2 \leftarrow \varnothing$
3. ***Input****: Assign LC to* C^1 *and* C^2 *and delete all LC courses fromsol*
4. **While** $(C^2 \neq \varnothing)$
5. $c \leftarrow$*Greedy_Course_Selection*(C^2, Sol)
6. $ATS \leftarrow$ Available_Timeslots(c, Sol)
7. **If** $(ATS = \varnothing)$
8. *Apply the Initial_Solution to assign courses of LC to Sol*
9. **End If**
10. Greedy_Timeslot_Selection(c, ATS)
11. Update(Sol)
12. $ATS \leftarrow$ Available_Timeslots(c, Sol)
13. **If** $(ATS = \varnothing)$
14. *Apply the Initial_Solution to assign courses of LC to Sol*
15. **End If**
16. Greedy_Timeslot_Selection(c, ATS)
17. Update(Sol)
18. $C^2 \leftarrow C^2 \backslash c$
19. **End While**
20. **While** $(C^1 \neq \varnothing)$
21. $c \leftarrow$ Greedy_Course_Selection (C^1, Sol)
22. $ATS \leftarrow$ Available_Timeslots(c, Sol)
23. **If** $(ATS = \varnothing)$
24. *Apply the Initial_Solution to assign courses of LC to Sol*
25. **End If**
26. Greedy_Timeslot_Selection(c, ATS)
27. Update(Sol)
28. $C^1 \leftarrow C^1 \backslash c$
29. **End While**
30. **Return** *Sol*

End Local_Search

4. Computational results

In this section, computational experiments are conducted to evaluate the performance of the solution approaches. The algorithms were coded in C++ using CPLEX 12.6.3 and run on a computer with an Intel Core i7 CPU and 40 GB of RAM. The following two subsections give details of the results attained using the proposed exact and metaheuristic methods, respectively.

4.1. Data sets generation and parameters setting

The tests were performed on 48 instances organized into 6 groups of 8 instances each. Table 4, in which the parameters of these sets are specified, includes a column defining the total number of variables each test set contains. As is evident from the table, the parameter $|Cst|$, which denotes the number of courses with non-deterministic status, has a significant impact on the number of variables because we have $|\Omega| = 3^{|Cst|}$ where number 3 is related to the three status of each course of set *Cst*, i.e. fixed, canceled and not-fixed. The values of parameters are randomly chosen, regarding the real-world data sets. Also, we set $\alpha = 0.2$ based on fine tunning.

Table 4
Parameters of the data sets.

Group	*wd*	*td*	*C*	C^1	C^2	$\|Cst\|$	*nf*	*lbs*	*lbc*	*NC*	$\|P\|$	*Number of variables*
1	3	3	10	5	5	2	2	3	3	2	4	2844
2	3	4	12	6	6	2	2	3	3	2	5	4716
3	4	3	14	7	7	3	3	3	3	2	5	15,561
4	4	4	16	8	8	3	3	3	3	2	6	24,198
5	4	4	16	8	8	4	3	3	3	2	6	71,826
6	5	4	22	11	11	4	3	3	3	2	9	126,291

In order to determine the parameters PF_{it} we followed the approach of Movahedfar, Ranjbar, Salari, and Rostami (2013) in which all courses are divided into four categories in terms of profitability of presentation each course in different time slots. For example, it is better for both students and professors to participate in difficult courses in the morning rather than time slots immediately after lunch time. Also, we consider identical values for weights W_1 to W_7in all of our developed test instances. In real cases, these values can be determined based upon the well-known Analytical Hierarchy Process (AHP) method (see Triantaphyllou, 2000) using a 7 × 7 matrix and expert judgment.

4.2. The results of the integrated model

Having solved the test problems, we provide the summarized results of the exact models in Table 5. As is presented, the optimal solution is obtained for 39 out of 48 instances as follows. The first four groups within the given time limit (3 h) are solved optimally. Considering the other two groups, 25% of the instances of group 5 are optimally solved in the determined time limit, while this value is 0% for the last group, which has the highest complexity among all. To further illustrate this issue, the running time, the CPLEX gap and the deviation of the optimality are pointed out in the following table.

Table 5
Results of the integrated model.

Group		1	2	3	4	5	6
Instance 1	Run time (s)	0.3	6.3	162.7	7601.6	10801.6	10802.2
	CPLEX gap	0	0	0	0	0.2	0.3
Instance 2	Run time (s)	0.4	1.1	3293.2	1910.8	10802.2	10802.8
	CPLEX gap	0	0	0	0	0.1	0.2
Instance 3	Run time (s)	0.8	1.8	76.2	2336.4	10801.9	10,800
	CPLEX gap	0	0	0	0	0.1	0.9
Instance 4	Run time (s)	1.3	1.3	332.1	2750.7	10800.6	10802.6
	CPLEX gap	0	0	0	0	0.2	0.3
Instance 5	Run time (s)	2.1	31.9	308.8	2078.3	10808.9	10,806
	CPLEX gap	0	0	0	0	0.1	0.3
Instance 6	Run time (s)	0.6	1.9	25.4	30.0	372.102	10802.1
	CPLEX gap	0	0	0	0	0.0	0.4
Instance 7	Run time (s)	1.6	1.4	127.6	295.9	2565.76	10802.4
	CPLEX gap	0	0	0	0	0.0	0.4
Instance 8	Run time (s)	10.6	7.3	563.4	949.7	10800.6	10,803
	CPLEX gap	0	0	0	0	0.9	0.3
Average run time		2.2	6.6	611.2	2244.2	8469.2	10802.6

4.3. EVPI index

Birge and Louveaux (2011) addressed the expected value of perfect information (EVPI) as a parameter measuring the maximum amount a decision maker would be ready to pay in return for complete and accurate information about the future. Regarding the stochastic programing, this parameter measures how much it is reasonable to pay to obtain perfect information about the future in case of uncertainty. In the problems like ours, in which uncertainty is supposed to be modeled through a number of scenarios with different probabilities, we are able to gain the value of having the complete information, i.e. where no uncertainty is included, using this parameter.

In order to calculate EVPI, we first consider a specific scenario which is certain to happen in the future, and then obtain the solution for that situation. Considering all the scenarios separately as the only

single happening scenario, there are several optimal solutions resulting in an identical objective value. Using the probabilities of scenarios as the weights, we then calculate the weighted sum of the objective values, which denotes the optimal value for the case where no uncertainty is involved. For example, such value in the instance #27 is −0.418. Having also the optimal solution obtained by solving the integrated model which is −0.396 for instance #27, we can reach the EVPI value. For instance #27, we have EVPI = −0.418 − (−0.396) = 0.021. In other words, for a minimization problem, the objective value reached by considering deterministic situation is better than the value obtained in case of uncertain parameters. The deviation of the integrated model from the optimal solution in case of certainty is equal to $\frac{-0.021}{-0.418} = 0.05$ or 5%. Similarly, we calculated an average of deviation over 34 instances, for ones the optimal solution of the integrated model was available, and the 4% result confirms the efficiency of the proposed integrated model.

4.4. Comparison of the integrated and separated models

In this section, the discussion is about performance comparison of the integrated model and the separated one. For this purpose, the 29 instances those solved and reached optimality using the integrated model, are separately solved by M1/M2, so that the M1 model is firstly implemented and then M2 is applied over the output of and M1 using Montecarlo simulation approach. For this, the M2 model is replicated as many as 100 $|\Omega|$ to give a high chance of realization to all scenarios. In order to compare the integrated and separated models, we define the percent deviation for instance q as $PD_q = \frac{z_q - z_q^*}{|z_q^*|} \times 100$ where z_q indicates the objective function obtained for instance q using a specified method and z_q^* shows the best found objective function for that instances among all developed solution approaches. Also, we define the *APD* as the average of *PD* over a group of instances.

Table 6 shows the results obtained from the mentioned process. The 34 problems ranked in descending order belong to the 5 groups of test sets, from 1 to 5 sequentially, where the last two problems are the only members of group 5. Focusing on the running time, it is evidently concluded that M1/M2 model runs in much less computing time in comparison to the integrated model. The reason is that M1/M2 model broke down a larger real-scale problem into two sub-problems, where a great number of decision variables are omitted due to inconceivable scenarios. It is to be mentioned that the average run time for a given instance is specified by adding the M1 running time to the mean of the computing times needed for the several runs of M2.

Also, Table 6 indicates that the average superiority percentage of the integrated model increases for larger size instances and also those instances having more course with uncertain status.

Furthermore, we investigate the performance of the developed approaches by applying them on some instances which differ in number of courses with uncertain status. In this regard, we choose 3 test problems where their optimal solution was previously gained in 3 hours time limit. Thereafter, 4 instances are generated by iteratively changing C_{st} in each of the specified problems, so the overall of 12 instances are created in from of 3 categories while the only difference in each group is C_{st}. For this part, instance 1 and 2 are chosen from group 4 and instance 3 is a member of group 3, where the parameter previously defined in Table 2. Having defined the 12 instances, we then solve them by both the integrated and separated models and report the obtained results in Table 7.

Table 7 indicates the superiority percentage of the integrated model grows as the number of uncertain courses ($|C_{st}|$) in each instance increases. In other words, the efficiency of the two-stage stochastic programming model approach is by far better in problems with the higher levels of uncertainty.

Table 6
Comparative results of the integrated and separated models.

Instance	Average run time of M1/M2	Run time of the integrated model	PD of M1/M2	Superiority percentage of the integrated model	Average of superiority percentage
1	0.32	0.28	0%	33%	42%
2	0.29	0.40	1%	26%	
3	0.32	0.82	4%	40%	
4	0.32	1.32	27%	59%	
5	0.36	2.14	41%	49%	
6	0.28	0.60	1%	21%	
7	0.35	1.56	9%	67%	
8	0.35	10.56	11%	39%	
9	0.74	6.35	7%	69%	50%
10	0.33	1.12	0%	44%	
11	0.42	1.79	2%	51%	
12	0.42	1.26	2%	24%	
13	0.70	31.86	0%	38%	
14	0.42	1.85	219%	50%	
15	0.38	1.42	11%	65%	
16	0.64	7.32	56%	59%	
17	0.51	162.74	0%	45%	54%
18	4.94	3293.22	72%	46%	
19	1.02	76.21	92%	71%	
20	0.79	332.14	9%	62%	
21	1.36	308.85	200%	40%	
22	0.91	25.38	0%	46%	
23	4.47	127.65	291%	45%	
24	1.19	563.44	40%	80%	
25	9.20	7601.63	41%	70%	56%
26	2.18	1910.77	0%	40%	
27	1.71	2336.41	24%	57%	
28	16.15	2750.66	35%	53%	
29	1.37	2078.31	0%	63%	
30	0.59	30.04	0%	43%	
31	2.32	295.96	5%	60%	
32	1.48	949.74	3%	58%	
38	0.50	372.10	3%	56%	58%
39	2.66	2565.76	76%	60%	
Average	1.8	760.3	38%	51%	

Table 7
Impact of the number of courses with uncertain status on the integrated and separated models.

$\|C_{st}\|$		1	2	3	4
Instance 1	Run time of the integrated model (s)	1	5.2	38.5	372.1
	Run time of M1/M2 (s)	0.5	0.3	0.4	0.4
	Superiority percentage of the integrated model	22%	34%	56%	65%
Instance 2	Run time of the integrated model (s)	6.0	37.5	269.3	2565.8
	Run time of M1/M2 (s)	3.7	5.1	3.8	3.6
	Superiority percentage of the integrated model	36%	39%	50%	65%
Instance 3	Run time of the integrated model (s)	2.3	8.9	91.8	2008.2
	Run time of M1/M2 (s)	1.3	1.1	1.1	0.9
	Superiority percentage of the integrated model	17%	63%	70%	71%
Average of superiority percentage of the integrated model		25%	45%	59%	67%

4.5. Detailed results of HA/M2 solution approach

In this section, we investigate the performance of HA/M2 approach. For this purpose, we implement both the integrated model and the HA/M2 approach with four different time limits. We impose a limit of 1, 10, 30 and 180 min on the CPU run times for the data sets and run the two proposed approaches within each time limit.

Table 8
Comparison of the integrated model and HA/M2.

Time limit (min)	1	10	30	180
Integrated model	406%	93%	51%	29%
HA/M2 model	387%	335%	284%	225%
Superiority percentage of the integrated model	71%	73%	79%	83%

Table 9
Comparison of the integrated model and HA/M2 based on the 6th group of instances.

Time limit (min)	1	10	30	180
Integrated model	1186%	361%	150%	47%
HA/M2 model	403%	212%	147%	92%
Superiority percentage of the integrated model	0%	4%	5%	6%

Table 8 provides the results of *APD* obtained by implementing the integrated model as well as the HA/M2 method, while imposing the mentioned time limits. As is given, both algorithms converge to optimal solutions in case of expanded running times. It is noteworthy that the HA/M2 approach is able to provide better results in 1-minute time limit, while underperforms the integrated model in larger running time limits. It is to be noted that in this table, the last row defines the superiority of integrated model over HA/M2 in percentage.

As regards the M1/M2 approach, the *APD* is 51% in 1-minute time limit. Since this method obtained the optimal solution of M1 for all the instance, no improvement has been achieved within the larger time limits. Although the results denote the better performance of this method rather than the integrated one (in 1-minute time limit), the integrated model attains better solutions in other time limits. It should be mentioned that regarding a 180-minute time limit, this algorithm has reached the value of 29% in terms of deviation percentage, where the integrated model is unable to show such level of performance.

As previously mentioned in the above table, the HA/M2 approach failed to outperform the integrated model in many instances. Since this method has been developed for large-size problems, we shift our focus on some instances of group 6 that did not reach the optimal solution in 180 min. Presenting Table 9, in which the *APD* values are summarized, we can conclude that the HA/M2 model outperforms the integrated model in large size instances, where the imposed time limit is up to 30 min. Regarding the superiority percentage of the integrated model in last row of Table 9, it can be concluded that HA/M2 model obtains better solutions in large-scale problems. It is to be mentioned that the value of *APD* for the large instances solved by M1/M2 method is 95%.

5. Conclusions and future research

In this paper, a university course timetabling problem has been studied while considering the cancelations risk of the provided courses. In many universities where the registration is implemented in two steps of (1) pre-registration and (2) drop/add phases, there may be some courses need to be canceled because of not reaching the required threshold in terms of total registered students. As a result, the cancelation risk accounts for the inherent uncertainty within the proposed problem, in which the satisfaction of students and professors are to be maximized. Regarding the solution method, we have developed a two-stage stochastic programming approach and compared the results with the M1/M2 model, where no uncertainty has been considered, as well as a heuristic method named HA/M2. Having investigated the obtained results, we reach to this point that the two-stage stochastic programming approach shows a good performance, especially in case of dealing with a high-level uncertainty. The HA/M2 model has been also able to provide qualified solutions for large-size instances in a limited CPU run time. The results obtained from the M1/M2 approach has shown that the solution obtained by this method can be considered as an appropriate initial solution when there is a moderate level of uncertainty in the problem, otherwise it fails to present qualified solutions.

For future research, we believe the following directions to be promising.

(I) In addition of cancelation of some courses, as a recourse action in the drop/add phase, it would be much practical to consider the possibility of adding some new courses from a limited number of not offered courses.
(II) In this model, the probabilities of scenarios have been assumed to be independent while they may inherit some levels of dependency. There is a high chance that a fewer registered students in one course be under the influence of a higher number of registered students in another course. As a research direction, it is an interesting topic and the Bayesian inference may help in this case.
(III) The proposed solution approach in this problem can provide good solutions for large size instances for those CPLEX failed to reach the optimality within a three-hour time limit. As a future study, developing new heuristic or metaheuristic solution approaches to obtain high-quality solutions for all types of instances is greatly beneficial.

Acknowledgement

This work is supported by Ferdowsi University of Mashhad as a research project with number 45028 and date 5-10-2017.

References

Abdullah, S., & Turabieh, H. (2012). On the use of multi neighbourhood structures within a Tabu-based memetic approach to university timetabling problems. *Information Sciences, 191*, 146–168.
Alvarez-Valdes, R., Crespo, E., & Tamarit, J. M. (2002). Design and implementation of a course scheduling system using Tabu Search. *European Journal of Operational Research, 137*(3), 512–523.
Asham, G. M., Soliman, M. M., & Ramadan, A. R. (2011). Trans genetic coloring approach for timetabling problem. *Artificial Intelligence Techniques Novel Approaches & Practical Applications IJCA,* 17–25.
Asratian, A. S., & de Werra, D. (2002). A generalized class–teacher model for some timetabling problems. *European Journal of Operational Research, 143*(3), 531–542.
Babaei, H., Karimpour, J., & Hadidi, A. (2015). A survey of approaches for university course timetabling problem. *Computers & Industrial Engineering, 86*, 43–59.
Bakir, M. A., & Aksop, C. (2008). A 0–1 integer programming approach to a university timetabling problem. *Hacettepe Journal of Mathematics and Statistics, 37*(1), 41–55.
Birge, J. R., & Louveaux, F. (2011). *Introduction to stochastic programming.* Springer Science & Business Media.
Cacchiani, V., Caprara, A., Roberti, R., & Toth, P. (2013). A new lower bound for curriculum-based course timetabling. *Computers & Operations Research, 40*(10), 2466–2477.
Cambazard, H., Hebrard, E., O'Sullivan, B., & Papadopoulos, A. (2012). Local search and constraint programming for the post enrolment-based course timetabling problem. *Annals of Operations Research, 194*(1), 111–135.
Carter, M. W., & Laporte, G. (1997). Recent developments in practical course timetabling. *International conference on the practice and theory of automated timetabling, PATAT97* (pp. 3–19). .
Carter, M. W., Laporte, G. G., & Lee, S. Y. (1996). Examination timetabling: Algorithmic strategies and applications. *Journal of the Operational Research Society, 47*, 373–383.
Ceschia, S., Di Gaspero, L., & Schaerf, A. (2012). Design, engineering, and experimental analysis of a simulated annealing approach to the post-enrolment course timetabling problem. *Computers & Operations Research, 39*(7), 1615–1624.
Chaudhuri, A., & De, K. (2010). Fuzzy genetic heuristic for university course timetable problem. *International Journal of Advanced Soft Computing and its Applications, 2*(1), 100–121.
De Werra, D. (1985). An introduction to timetabling. *European Journal of Operational Research, 19*(2), 151–162.
Goh, S. L., Kendall, G., & Sabar, N. R. (2017). Improved local search approaches to solve the post enrolment course timetabling problem. *European Journal of Operational Research, 261*(1), 17–29.
Gotlieb, C. C. (1963). The construction of class-teacher timetables. *IFIP congress* (pp. 73–77). .
Henry Obit, J. (2010). *Developing novel meta-heuristic, hyper-heuristic and cooperative search for course timetabling problems* (Doctoral dissertation)University of Nottingham.
Khonggamnerd, P., & Innet, S. (2009). On improvement of effectiveness in automatic university timetabling arrangement with applied genetic algorithm. *Computer sciences and convergence information technology. ICCIT'09. Fourth international conference on*

(pp. 1266–1270). IEEE.
Kim, K., & Mehrotra, S. (2015). A two-stage stochastic integer programming approach to integrated staffing and scheduling with application to nurse management. *Operations Research, 63*(6), 1431–1451.
Lewis, R. (2008). A survey of metaheuristic-based techniques for university timetabling problems. *OR Spectrum, 30*(1), 167–190.
Lewis, R., Paechter, B., & McCollum, B. (2007). *Post enrolment based course timetabling: A description of the problem model used for track two of the second international timetabling competition.* Cardiff Business School, Technical Report.
Lewis, R., & Thompson, J. (2015). Analysing the effects of solution space connectivity with an effective metaheuristic for the course timetabling problem. *European Journal of Operational Research, 240*(3), 637–648.
Méndez-Díaz, I., Zabala, P., & Miranda-Bront, J. J. (2016). An ILP based heuristic for a generalization of the post-enrollment course timetabling problem. *Computers & Operations Research, 76*, 195–207.
Movahedfar, N., Ranjbar, M., Salari, M., & Rostami, S. (2013). Memetic and scatter search metaheuristic algorithms for a multi-objective fortnightly university course timetabling problem: A case study. *Journal of Industrial and System Engineering, 6*(4), 249–271.
Song, T., Liu, S., Tang, X., Peng, X., & Chen, M. (2018). An iterated local search algorithm for the University Course Timetabling Problem. *Applied Soft Computing, 68*, 597–608.
Triantaphyllou, E. (2000). *Multi-criteria decision making methods: A comparative study.* Springer.
Tuga, M., Berretta, R., & Mendes, A. (2007). A hybrid simulated annealing with kempe chain neighborhood for the university timetabling problem. *Computer and information science, 2007. ICIS 2007. 6th IEEE/ACIS international conference on* (pp. 400–405). IEEE.
Van den Broek, J. J. J., & Hurkens, C. A. (2012). An IP-based heuristic for the post enrolment course timetabling problem of the ITC2007. *Annals of Operations Research, 194*(1), 439–454.
Welsh, D. J., & Powell, M. B. (1967). An upper bound for the chromatic number of a graph and its application to timetabling problems. *The Computer Journal, 10*(1), 85–86.
Zadeh, L. A. (1965). Fuzzy Sets. *Information and Control, 8*, 338–353.