Teacher-oriented Fairness in Course Timetabling

Rostislav Burget · **Hana Rudová**

Abstract Fairness introduces an important optimization criterion which needs to be handled at a reasonable level to generate acceptable timetables for particular entities. We propose an extension of the fairness measure which truly reflects the preferences of entities such as teachers or classical curricula. An incremental algorithm for the fairness measure computation is proposed for inclusion in iterative or constructive search algorithms. We demonstrate its inclusion into the iterative forward search algorithm. The implementation is available as a part of the complex UniTime system for course and examination timetabling. We apply our approach on real-life problems from Masaryk University and show improvements in fairness for teachers at a cost which is a very reasonable compromise with other objective functions. In addition, we demonstrate that a weighted inclusion of a fairness criteria allows us to achieve a proper balance with respect to other objective functions.

Keywords Course Timetabling · Fairness · Multi-objective approach · Timetabling system · UniTime · Real-world problem · Search

1 Introduction

Fairness plays an important role in acceptance of automated solutions by particular entities such as nurses, employees, or users. These entities need a proper balance in handling their preferences. Early studies of fairness come from the area of computer networks [3] in problems such as fair bandwidth allocation [14]. Particular entities may be nurses in nurse rostering [7,5] or employees in employee scheduling [16]. Another view comes from job scheduling, where fairness is related to jobs or to users who submitted their jobs [18,4]. In educational timetabling, fairness has been more deeply studied in past several years. Recent studies were related to the fairness of

R. Burget and H. Rudová
Faculty of Informatics, Masaryk University
Brno, Czech Republic
E-mail: hanka@fi.muni.cz

timetable for particular curricula [8] or for individual students [9]; however, we are not aware of any work where fairness for teachers has been be considered.

In this paper we will discuss a proposal for measuring fairness and present an implementation incorporating this measure as an extension to the UniTime[1] timetabling system which allows solution of large-scale course and examination timetabling problems. This system has been applied at many universities world-wide since its initial development for Purdue University [10, 13] in the USA. Our inspiration for further study of fairness measures comes from a European university, Masaryk University in the Czech Republic, where four faculties use UniTime for course timetabling. In the implementation at Masaryk University, teachers play a more important role than in USA, where many instructors may not be specified in the initial timetabling problem. Fair consideration of teaching schedules is an important problem directly related to the overall satisfaction with the generated timetables.

Our intent was to propose a proper fairness measure which could have been easily included into the existing timetabling system with its objective functions being weighted differently. It is important that the proposed extension to include a fairness measure can work directly on top of the preferences (or penalties) associated with particular assignments. Our approach takes into account the preferences associated with the actual assignment, as well as those associated with most desirable assignment, for a class and reflects them in the fairness measure computation. The computed fairness measure can then be included into a standard objective function with weighted criteria. An algorithm for incremental computation of fairness is proposed and implemented as a part of UniTime system. Experiments on real-life data sets from Masaryk University show a significant improvement in fairness, and in time preferences, at a reasonable cost versus other criteria. This comparison is presented with respect to timetables generated by the standard UniTime system without a fairness component in the objective function. Real-life data sets are used to introduce real-life problems with 500 – 600 classes (timetabled events of courses) and 1,700 – 1,900 students with 17,000 – 20,000 course enrollments. About 240 teachers with an interest in fair assignment of class times are included.

The structure of our paper is as follows. The next section describes the objective function with multiple criteria but with no fairness criteria included. Sec. 3 discusses various fairness measures available in the literature. Sec. 4 introduces our proposal of teacher-oriented fairness and Sec. 5 proposes a proper incremental algorithm for the fairness measure computation. Sec. 6 presents our experimental problems and results. The last section provides conclusions resulting from this work.

2 Objective function

The classical approach to solving timetabling problems [13] presents an overall objective function to be minimized as a weighted sum of particular objectives depending

[1] `http://www.unitime.org`

on the selected solution s, i.e.,

$$F(s) = \sum_{k=1}^{m} w_k \cdot F_k(s) \quad . \tag{1}$$

In course timetabling, particular objective functions may represent time preferences, room preferences, or the number of student conflicts to be minimized. Student conflicts can be established directly by students who individually enroll into particular courses or by students who were generated in a manner that respects an existing curriculum [12]. We may also need to represent additional soft constraints among classes (e.g., a lecture class should precede seminar classes, classes should be taught sequentially, a gap among classes is preferred). Given these, we may have the objective function

$$F(s) = w_{stud} F_{stud}(s) + w_{time} F_{time}(s) + w_{room} F_{room}(s) + w_{distr} F_{distr}(s) \tag{2}$$

where F_{stud}, F_{time} and F_{room} denote student conflicts, time preferences, and room preferences, respectively. F_{distr} is reserved for those additional soft constraints which are called distribution preferences in UniTime. All objective functions are weighted by their corresponding weights.

3 Fairness Measures

Fairness can be considered as an additional component of an objective function. In this case, the fairness measure corresponds to one of the $F_k(s)$ in Eqn. 1 and it should be weighted by a fairness weight w_{fair} to handle its importance appropriately and possibly normalize it with respect to other criteria.

The fairness of the solution s depends on the set of penalties $P_i(s)$ of particular entities i for which a fair distribution is necessary. In course timetabling, $P_i(s)$ may correspond to a penalization for each teacher i or for each curriculum i. However, it is important to realize that curricula fairness is not so easy to achieve for more complex timetabling problems. As mentioned, we need to compute a penalty $P_i(s)$ for each entity i. This can be simply done for teachers as we include penalty information about all classes (events of the courses) taught by each teacher. It is not so simple for curricula with elective courses and course sections [12], where students take various subsets of the classes which are related to one curriculum. In such a complex structure it is hard to relate fairness of the students directly to the curricula. So, we leave this consideration for future work.

A common fairness measure used in many areas is the Jain's fairness index [3]. It has been proposed such that it is independent of size, scale, and units. Its value always lies between 0 (complete unfairness) and 1 (total fairness). In addition, it is continuous, meaning that any change in resource allocation is related to a change in the index.

Jain's fairness index corresponds to

$$P^{Jain}(s) = \frac{(\sum_{i=1}^{n} P_i(s))^2}{n \cdot \sum_{i=1}^{n} P_i(s)^2} \quad . \tag{3}$$

Unfortunately this index is not always proper as a fairness measure because it is dependent on the relative sizes of individual penalties $P_i(s)$. Generally a higher value of Jain's index may not mean a better fairness, but it can be related to a uniform deterioration for all penalties [5].

A base fairness measure minimizes the quality of the worst individual penalty.

$$P^{Max}(s) = \max_{i \in \{1,\dots,n\}} P_i(s) \tag{4}$$

Computation of individual penalty may include various types of penalties weighted by its importance or cost as it was done in [15]. For inclusion into the overall objective $F(s)$ (Eqn. 1), it could be multiplied by the number of entities n [7,5]. However, this approach is troublesome when there is some entity with very bad penalty which cannot be improved. In this case, as it is common for all min-max approaches, a bad case does not allow for comparison of other penalties. Also two solutions with the same worst case penalty cannot be distinguished.

Other fairness measures [5,7] can extend it by consideration of both the worst and the best case

$$P^{Error}(s) = \max_{i \in \{1,\dots,n\}} P_i(s) - \min_{i \in \{1,\dots,n\}} P_i(s) \tag{5}$$

or by consideration of the lexicographic approach [8,5]. In this case, permutation of all individual penalties $P_{i_j}(s)$ sorted in a non-increasing order is taken as a fairness measure of solution s

$$P^{Lex}(s) = (P_{i_1}(s), P_{i_2}(s), \dots, P_{i_n}(s)) \quad \text{s. t.} \quad P_{i_1}(s) \leq P_{i_2}(s) \leq \dots \leq P_{i_n}(s) \ . \tag{6}$$

Next, the best (and also the better) solution is obtained by a lexicographic ordering of penalties $P^{Lex}(s)$ for all solutions s. This approach would certainly need some other reasoning than a simple inclusion into the overall objective function, e.g., a multi-criteria optimization.

Other classical approach is to minimize the sum of squares of particular penalties

$$P^{SS}(s) = \sqrt{\sum_{i=1}^{n} P_i(s)^2} \tag{7}$$

which emphasizes minimization of very bad cases. It is sometimes presented in the form [5] where squares of other objective function(s) should be also under the same root. In any case, (combined) squares of weights easily introduce very different criteria and must be taken with care.

Deviations from the average penalty discussed in [5] has also been proposed as a criteria for evaluating the fairness of candidate solutions

$$P^{Dev}(s) = \sum_{i=1}^{n} \left| P_i(s) - \frac{1}{n} \sum_{j=1}^{n} P_j(s) \right| \ . \tag{8}$$

Our extension further specifies how particular penalties $P_i(s)$ should be computed (see next section). This approach allows us to take into account all entities (not just the worst or the best) and can be simply included as a weighted component of the overall objective function such as Eqn. 1. Last but not least, there is no necessity to transform weights which could be directly taken from Eqn. 1 as we will see below. Given that, we have decided to consider this type of fairness for inclusion in our work.

4 New Teacher-oriented Fairness

We now propose a new fairness measure which we apply to construct a fair timetable for teachers. This same measure can also be applied when fairness for curricula should be achieved as in [8]. In this case, classical curricula containing a set of courses may be considered in correspondence with the curriculum-based timetabling problem from the second international timetabling competition ITC 2007 [2].

For now we will concentrate on fairness among teachers. First, we need to compute the *actual penalty* $\overline{P_i}(s)$ for each teacher i using penalties $p_k(j,s(j))$ for all of his/her classes j with the placement $s(j)$. Also we will use notation $\overline{P}(s)$ to refer the actual penalty for all teachers i. Generally we can try to be fair with respect to all m objective functions having their weights w_k (Sec. 2)

$$\overline{P_i}(s) = \sum_{j=1}^{n_i} \sum_{k=1}^{m} w_k p_k(j,s(j)) \tag{9}$$

for teacher i having n_i classes. Note that s is understood as a function which gives for each variable/class j its value/placement $s(j)$. Mostly we are interested in fairness related to some criteria only. A particular penalty k might be a time preference and a room preference, which would mean that $m = 2$.

$$\overline{P_i}(s) = \sum_{j=1}^{n_i} \left(w_{time} p_{time}(j,s(j)) + w_{room} p_{room}(j,s(j)) \right) \quad . \tag{10}$$

Alternatively, we might just consider time preferences as the most important and concentrate on fairness of the class' times for particular teachers. For this case, we obtain the actual penalty

$$\overline{P_i}(s) = \sum_{j=1}^{n_i} w_{time} p_{time}(j,s(j)) \quad . \tag{11}$$

Penalty $p_{time}(j,s(j))$ would then be just a time preference corresponding to the assignment of the class j to the placement $s(j)$ in time.

Next, we compute the *best penalty* $\underline{P_i}$ for the teacher i

$$\underline{P_i} = \min_a \overline{P_i}(a) \tag{12}$$

which can be achieved for any possible assignment a [2]. This can be easily computed based on the best time and/or room preferences available for each class because we can simply take sum of the best preferences (rather than minimize sum of preferences). The notation $\underline{P}$ then refers to the best penalties for all teachers i.

Based on the best and actual penalty, we can compute the *final penalty* $P_i(s)$ for a teacher i with n_i classes

$$P_i(s) = \left(\overline{P_i}(s) - \underline{P_i} \right) / n_i \quad . \tag{13}$$

[2] Note that assignments are just all possible value combination for all variables, they do not need to satisfy any constraints.

Again $P(s)$ denotes the final penalties for all teachers i.

This final penalty may be applied in any fairness measure defined in Sec. 3. More specifically, we will describe an efficient algorithm for the deviation-based fairness measure $P^{Dev}(s)$. In this case, the objective function can be an extended version of Eqn. 2

$$\begin{aligned} F(s) = & \; w_{stud}F_{stud}(s) + w_{time}F_{time}(s) + \\ & + w_{room}F_{room}(s) + w_{distr}F_{distr}(s) + w_{fair}P^{Dev}(s) \quad . \end{aligned} \tag{14}$$

We will demonstrate some of the important features of the final penalty in the following example.

Example 1 Consider the teacher i who has 3 classes. The first class in our solution s has a preferred time assignment with a preference $p_{time}(1,s(1)) = -4$ and discouraged time assignments having the preference 4. Two other classes do not have any preferences, i.e., their time preferences correspond to 0. Next, we select the preferred value for the first class and any of values for the remaining classes. In this case, the best penalty is $\underline{P_i} = -4+0+0$ and the actual penalty is $\overline{P}_i(s) = 4+0+0$. The final penalty corresponds to $P_i(s) = (4-(-4))/3 = 2.6\overline{6}$.

We can see that our fairness measure is simply established on preferences available in the problem definition. Taking into account the best penalty allows us to differentiate cases when teachers have different expectations about preferences of their classes. We just compare the desire of teacher (reflected in the best penalty) with the achieved quality (reflected in the actual penalty). In addition, taking an average value for all classes of one teacher allows us to obtain comparable values for final penalties which can be directly included in the fairness measures presented in Sec. 3.

5 Incremental Algorithm

To more easily understand the process of incremental computation of the fairness measure, we present our algorithm as a part of the iterative forward search procedure (IFS) [13] which is applied in UniTime for iterative computation of the solution (Fig. 1). It demonstrates how the fairness measure is incrementally computed during the search. Nonetheless a similar incremental computation can be directly applied in other iterative or constructive algorithms such as local search algorithms [17] or look ahead search [1].

We first explain the basic iterative forward search as it is written in blue (light in black/white print). The algorithm runs in cycles (Ln. 8–18) with the goal to improve the quality of the current timetable/solution s and to return the best solution s_0 at the end (Ln. 19). In each cycle, one class/variable is selected (Ln. 10) as well its new placement/value (Ln. 11). This new assignment v/d may possibly conflict with some existing assignments which are computed in Ln. 12. These are removed and the class v is assigned its new placement d (Ln. 13). This is certainly just the base structure of the algorithm, more details about fundamental heuristics involved can be found in [10,13].

```
1: function FAIRIFS(𝒫, F)     {problem 𝒫, objective function F}
2:    j = 0
3:    s = ∅                         {current timetable}
4:    s_o = ∅                       {the best timetable}
5:    compute = false
6:    P^Dev(s) = 0
7:    P̲ = COMPUTEBESTPENALTY(𝒫)
8:    while CANCONTINUE(s, j) do
9:        j = j + 1
10:       v = SELECTVARIABLE(𝒫,s)          {class v to be assigned}
11:       d = SELECTVALUE(𝒫,s,F,v)         {placement d to be assigned to v}
12 :      Y = HARDCONFLICTS(𝒫,s,v/d)       {classes to be unassigned in Y}
13 :      s = s\Y ∪ {v/d}                  {update current timetable s}
14:       if compute then P^Dev(s) = INCREMENTFAIRNESS(𝒫, s, v, d, Y)
15:       else if s is complete then
16:           compute = true
17:           P^Dev(s) = COMPUTEFAIRNESS(𝒫, s)
18:       if F(s) < F(s_o) then s_o = s
19:    return s_o
```

Fig. 1 Fairness measure computation within the iterative forward search (the blue/light text represents the original IFS, the black text presents a new fairness measure computation)

Now we can describe how to compute the fairness measure (see the black rows in Fig. 1). Input of the algorithm remains the same, the function F now includes in its sum the component for the fairness measure $w_{fair}P^{Dev}(s)$ (Eqn. 14). Initially the fairness measure $P^{Dev}(s)$ is set to 0 (Ln. 6). This value is used to compute $F(s)$ before the current timetable s is complete because the fairness measure may be rather unrealistic before all classes are assigned. Next we can compute for all teachers i the best penalty $\underline{P_i}$ (Ln. 7) because it remains the same through all computations (detailed description given later along with Fig. 2). The fairness measure $P^{Dev}(s)$ is first computed when a complete solution s is found (Ln. 15–17). In following iterations, the fairness measure is incrementally maintained only (Ln. 14). Detail descriptions of both functions will be given subsequently.

The function computing the best penalty is presented in Fig. 2. In summary, the best penalty for the teacher i is the sum of the smallest penalties for all his/her classes. For each class v, we compute the best (smallest) penalty (Ln. 3–7). It is computed

```
1: function COMPUTEBESTPENALTY(𝒫)
2:    forall teachers i in 𝒫 do P̲_i = 0
3:    forall classes v in 𝒫
4:        p = ∞
5:        forall placements d of v do
6:            q = ∑_k w_k p_k(v,d)
7:            if q < p then p = q
8:        forall teachers i of class v do P̲_i = P̲_i + p
9:    return P̲
```

Fig. 2 Function for computation of the best penalty $\underline{P_i}$ for all teachers i

such that all possible placements of the class are scanned (Ln. 5–7), the penalty is retrieved for each of them and the smallest value is maintained in p. In the next step, the computed smallest class penalty p is added to the best penalties of all teachers i of the current class v (Ln. 8).

The function computing the actual penalty for the first complete solution s is presented in Fig. 3. These initial actual penalties $\overline{P}(s)$ are computed in a similar manner

1: **function** COMPUTEACTUALPENALTY($\mathscr{P}$, s)
2: **forall** teachers i in $\mathscr{P}$ **do** $\overline{P_i}(s) = 0$
3: **forall** classes v in $\mathscr{P}$
4: let $v/d \in s$
5: $p = \sum_k w_k p_k(v,d)$
6: **forall** teachers i of class v **do** $\overline{P_i}(s) = \overline{P_i}(s) + p$
7: **return** $\overline{P}(s)$

Fig. 3 Function for computation of the best penalty $\underline{P_i}$ for all teachers i

to the best penalty $\underline{P}$ (Fig. 2) where we just take the penalty of the current class placement v/d (Ln. 4–5).

Fig. 4 explains initial computation of the fairness measure. Given the actual penal-

1: **function** COMPUTEFAIRNESS($\mathscr{P}$, s)
2: $\overline{P}(s) = $ COMPUTEACTUALPENALTY($\mathscr{P}$,s)
3: **forall** teachers i with n_i classes **do** $P_i(s) = \left(\overline{P_i}(s) - \underline{P_i}\right)/n_i$
4: $P^{avg} = \frac{1}{n}\sum_{i=1}^{n} P_i(s)$
5: **return** $\sum_{i=1}^{n} |P_i(s) - P^{avg}|$

Fig. 4 Initial computation of the fairness measure

ties $\overline{P}(s)$ from Ln. 2, we can compute the final penalty $P_i(s)$ for each teacher i (Ln. 3). In the next step, the average value of the final penalty P^{avg} among all teachers is computed (Ln. 4) and it is consequently applied in computation of the fairness measure $P^{Dev}(s)$ to be returned (Ln. 5).

After the fairness measure is initially computed it is iteratively maintained during the search (Fig. 1) by function in Fig. 5. We take into account the fact that the fairness measure can be changed only due to the distinct assignment of one or more classes which were in the given search step assigned (one class denoted by v, Ln. 10, Fig. 1) and unassigned (from none to several classes taking a part in Y, Ln. 12, Fig. 1). This means that the final penalty value can be only changed for teachers who participate in the changed classes. More precisely, when a class v is assigned a new placement d, its penalty is *added* to the actual penalties of all teachers of the class (Ln. 3) and the final penalty is updated for these teachers (Ln. 4). A similar computation is completed when some classes are unassigned, we just need to handle changes for teachers of all these classes. In this case, the actual penalties are *decreased* by a penalty for the placement g of unassigned class c (Ln. 6) and the corresponding final penalties are updated (Ln. 7). The average final penalty P^{avg} is computed in Ln. 8. No incremental

1: **function** INCREMENTFAIRNESS($\mathscr{P}, s, v, d, Y$)
2: **forall** teachers i of class v (i has n_i classes) **do**
3: $\overline{P_i}(s) = \overline{P_i}(s) + \sum_k w_k p_k(v,d)$
4: $P_i(s) = \left(\overline{P_i}(s) - \underline{P_i}\right)/n_i$
5: **if** $Y \neq \emptyset$ **then forall** teachers i of classes c/g in Y **do**
6: $\overline{P_i}(s) = \overline{P_i}(s) - \sum_k w_k p_k(c,g)$
7: $P_i(s) = \left(\overline{P_i}(s) - \underline{P_i}\right)/n_i$
8: $P^{avg} = \frac{1}{n}\sum_{i=1}^{n} P_i(s)$
9: **return** $\sum_{i=1}^{n} |P_i(s) - P^{avg}|$

Fig. 5 Function incrementally recomputing the fairness measure

computation is necessary in this step, it is not computationally demanding. Incremental computation would actually introduce rounding errors which is undesirable. Last but not least, the fairness measure $P^{Dev}(s)$ is computed and returned in Ln. 9.

6 Evaluation

6.1 Data Sets

Real-life data from the Faculty of Informatics at Masaryk University has been used in all experiments. Particular data instances are described in Table 1 where we can see the most important data attributes. For each of the four semesters, there is information about the number of classes (timetabled events of courses), teachers, rooms[3] and students. Student pre-enrollments play an essential role in constructing the timetable, i.e., the enrollment-based timetabling problem [6] has been solved and the number of enrollments is presented.

Table 1 Parameters of particular data sets

Semester	Classes	Teachers	Rooms	Students	Enrollments
Spring 2014	500	250	31	1,773	17,581
Autumn 2014	594	240	32	1,857	20,520
Spring 2015	511	229	28	1,650	16,845
Autumn 2015	603	247	34	1,894	20,003

The fairness measure was applied to construct fair timetables with respect to times of classes for particular teachers because the time is always a very important characteristic for teachers at the faculty. This means that the fairness measure relies on the actual penalty computed using Eqn. 11.

6.2 Results

Experiments have been run on a machine with dual core i5 2.5 GHz processor and 8 GB of RAM, using MS Windows 7 Professional, JDK 1.7 and CPSolver 1.3.58

[3] Changing number of rooms is due to the reconstruction completed just before the autumn 2015.

without the support for parallel solver runs (one thread has been used). Average results of the 10 runs each taking one hour are presented.

6.2.1 Different Fairness Weight

The first experiment presented in Table 2 shows changes in the fairness measure P^{Dev} based on the changes of the fairness weight w_{fair}. Evaluation also shows percentile satisfaction of time preferences F_{time}, room preferences F_{room}, distribution preferences F_{dist} and the number of student conflicts F_{stud}. Results for this experiment are computed for the Autumn 2015 data instance having the largest number of classes.

Table 2 Experiments with different fairness weight for Autumn 2015

w_{fair}	F_{time} (%)	F_{stud}	F_{dist} (%)	F_{room} (%)	P^{Dev} (%)
–	77.30	877.7	91.72	74.30	–
1	81.34	868.4	92.10	71.32	-14.7
3	84.66	905.1	90.95	71.41	-25.3
5	87.94	919.3	90.21	72.23	-46.5
7	88.86	927.7	91.72	70.83	-53.0
9	90.30	946.6	91.00	70.66	-58.3

The first row of Table 2 shows results of the runs where no fairness is applied and the standard UniTime timetabling solver is used for the timetable construction. This serves as a basis for our comparisons. Other rows show results of the runs where fairness weight w_{fair} was subsequently strengthened to emphasize fairness more. It is good to see that increasing the fairness weight subsequently improves the fairness measure P_{Dev}. We can see the percentile improvements of this value with respect to the base experiment (first line) where fairness measure was computed but not optimized. However, we need to carefully check satisfaction of other criteria. It is not surprising that F_{time} is getting better with a bigger w_{fair} because the fairness is established on the fairness for time of classes. The consequence is an improvement of F_{time} as well. While improving P^{Dev} and F_{time}, other criteria are shown to worsen, this is always the cost of improvement in the quality of remaining criteria. Selection of the best weight would always be related to the policies of particular school or university and implemented by the person responsible for timetable creation. We have decided to select $w_{fair} = 5$ for further experiments because satisfaction of criteria F_{stud}, F_{room} and F_{dist} is reasonable and there is a significant improvement of both P^{Dev} and F_{time}.

6.2.2 Results for All Data Sets

Having set the weight w_{fair} we can now present experimental results for all data instances. Results are available in Table 3 with the same structure as before. In addition, standard deviations are presented in the table. We can see that results are comparable for all data sets, with similar results as for the first experiment having been obtained. The fairness as well as time preferences are significantly improved compared to the run of the standard UniTime solver. Quality of remaining criteria was worse but the change still looks like a reasonable compromise.

Table 3 Results for all data sets with no fairness and with the fairness weight equal to 5

Semester	w_{fair}	F_{time} (%)	F_{stud}	F_{dist} (%)	F_{room} (%)	P^{Dev} (%)
Spring 2014	–	68.7±1.4	629.3±14.8	90.8±0.4	85.9±2.6	–
	5	81.5±1.5	660.4±13.6	91.3±1.4	84.8±1.9	-43.8±3.0
Autumn 2014	–	76.9±1.4	1,110.9±36.0	92.8±1.7	81.9±2.7	–
	5	86.7±1.0	1,179.1±46.1	87.6±1.8	81.9±2.3	-41.5±2.6
Spring 2015	–	76.2±1.2	853.9±12.4	91.0±2.0	88.8±2.0	–
	5	85.1±1.0	894.9±11.9	91.1±1.8	86.1±3.0	-40.0±3.4
Autumn 2015	–	77.3±1.3	877.7±22.5	91.7±1.2	74.3±5.4	–
	5	87.9±1.5	919.3±42.6	90.2±2.0	72.2±4.6	-46.5±11.6

7 Conclusion

Fairness plays an important role in acceptance of generated timetables. This has been realized in many recent studies with the main emphasis on fairness [8, 7, 5, 16]. Our approach allows us to establish fairness for entities in educational timetabling as well as in other problems where different preferences or penalties of entities are compared. To our knowledge, there are no earlier studies of fairness for teaches who were of the main interest in this work.

We proposed an incremental algorithm for inclusion of our fairness measure extension based on deviations. It can be included in any search with subsequent changes such as local search or backtracking-based search. Experiments on recent real-life data from Masaryk University in the Czech Republic show significant improvement in the proposed fairness measure at a reasonable cost in other criteria. Even more there is a significant improvement in the satisfaction of time preferences.

We included our algorithm into the UniTime system where it is available from version 4.1. It also means that source code of our implementation is fully available in this open source system as a part of the CPSolver library. The solution with the new fairness criteria will be presented to timetablers at universities using UniTime such as Masaryk University and Purdue University for potential use in practice.

In course timetabling, the most interesting future work is related to fairness for more complex entities, such as curricula, where the structure is not so clear. In the presence of course sections and elective courses, it is still an unsolved problem. Also it could be valuable to consider the fairness for individual students, which relates to student scheduling and sectioning problems [11]. Extension of the approach towards other problems where particular entities (nurses, employees, or users) have several different interests (their shifts, works, or jobs) could be certainly considered.

Acknowledgements We would like to thank to Tomáš Müller for his consultations and help with inclusion of the code into UniTime distribution.

References

1. Dechter, R.: Constraint Processing. Morgan Kaufmann Publishers (2003)
2. Di Gaspero, L., McCollum, B., Schaerf, A.: The second international timetabling competition (ITC-2007): Curriculum-based course timetabling (track 3). Tech. Rep. QUB/IEEE/Tech/ITC2007/-CurriculumCTT/v1.0, University, Belfast, United Kingdom (2007)
3. Jain, R.K., Chiu, D.M.W., Hawe, W.R.: A quantitative measure of fairness and discrimination for resource allocation in shared computer systems. Tech. Rep. DEC-TR-301, Digital Equipment Coporation (1984)
4. Klusáček, D., Rudová, H.: A metaheuristic for optimizing the performance and the fairness in job scheduling system. In: Artificial-Intelligence Applications in Information and Communication Technologies, pp. 3–29. Studies in Computational Intelligence Series 607, Springer (2015)
5. Komarudin, M.A.G., Smet, P., De Feyter, T., Vanden Berghe, G.: A two-phase heuristic and a lexicographic rule for improving fairness in personnel rostering. In: 10th International Conference on the Practice and Theory of Automated Timetabling, pp. 292–308 (2014)
6. Lewis, R., Paechter, B., McCollum, B.: Post enrolment based course timetabling: A description of the problem model used for track two of the second international timetabling competition. Cardiff Working Papers in Accounting and Finance A2007-3, Cardiff Business School, Cardiff University (2007)
7. Martin, S., Ouelhadj, D., Smet, P., Vanden Berghe, G., Özcan, E.: Cooperative search for fair nurse rosters. Expert Systems with Applications **40**, 6674–6683 (2013)
8. Mühlenthaler, M., Wanka, R.: Fairness in academic course timetabling. Annals of Operations Research **239**, 171–188
9. Muklason, A., Parkes, A.J., McCollum, B., Özcan, E.: Fairness in examination timetabling: Student preferences and extended formulations. In: 10th International Conference on the Practice and Theory of Automated Timetabling, pp. 512–515 (2014)
10. Müller, T.: Constraint-based timetabling. Ph.D. thesis, Charles University in Prague, Faculty of Mathematics and Physics (2005)
11. Müller, T., Murray, K.: Comprehensive approach to student sectioning. Annals of Operations Research **181**(1), 249–269 (2010)
12. Rudová, H., Müller, T.: Real-life curriculum-based timetabling with elective courses and course sections. Annals of Operations Research **239**, 153–170
13. Rudová, H., Müller, T., Murray, K.: Complex university course timetabling. Journal of Scheduling **14**(2), 187–207 (2011)
14. Salles, R.M., Barria, J.A.: Lexicographic maximin optimisation for fair bandwidth allocation in computer networks. European Journal of Operational Research **185**(2), 778–794 (2008)
15. Smet, P., Martin, S., Ouelhadj, D., Özcan, E., Vanden Berghe, G.: Investigation of fairness measures for nurse rostering. In: 9th International Conference on the Practice and Theory of Automated Timetabling, pp. 369–372 (2012)
16. Stockwell-Alpert, E., Chung, C.: Fairness in employee scheduling. In: MISTA 2015, Proceedings of the 7th Multidisciplinary International Conference on Scheduling: Theory and Applications, pp. 687–698 (2015)
17. Talbi, E.G.: Metaheuristics: From Design to Implementation. John Wiley & Sons, Inc. (2009)
18. Vasupongayya, S., Chiang, S.: On job fairness in non-preemptive parallel job scheduling. In: International Conference on Parallel and Distributed Computing Systems, PDCS 2005, pp. 100–105 (2005)