Computational Complexity Issues in University Interview Timetabling

Yuuki Kiyonari[1], Eiji Miyano[1⋆], and Shuichi Miyazaki[2⋆⋆]

[1]Department of Systems Innovation and Informatics,
Kyushu Institute of Technology, Fukuoka 820-8502, Japan
{kiyonari@theory., miyano@}ces.kyutech.ac.jp

[2]Academic Center for Computing and Media Studies,
Kyoto University, Kyoto 606-8501, Japan
shuichi@media.kyoto-u.ac.jp

1 Introduction

In the Department of Intelligence Science and Technology, Graduate School of Informatics, Kyoto University, there are approximately 20 professors, and 40 graduate course students in each year. In the final year of the course, every student submits a research thesis, and presents his/her work in twenty minutes to obtain a master degree. Presentation meeting is scheduled for two days, and in general, all professors attend the meeting and listen to all students' presentation. For each student, three professors (usually from the same department) are assigned as a referee basically according to the presentation topic and professors' research fields, and it is mandatory for referees to attend and evaluate the assigned students' presentation.

This presentation meeting used to be scheduled in one room, and there have been no serious problems. However, because of the increased number of graduate students, it became difficult to hold the meeting in two days this year, and we adopted a parallel session using two rooms. Then, there arises two major restrictions: (1) Two students evaluated by the same referee must be assigned to different timeslots. (2) Professors want to minimize the number of movements between rooms. Restriction (1) is a hard constraint, and the problem requires to decide if a feasible schedule exists. It is easy to see that this problem can be solved in polynomial time, and hence in this paper we focus on restriction (2), which is a soft constraint: We want to find a feasible schedule which minimizes the total number of movements of all professors.

To understand the problem, consider the following small example: There are six students s_1 through s_6 and six professors p_1 through p_6. The assignment of professors to students is illustrated in Fig. 1: Student s_1 is evaluated by two professors p_1 and p_2, and so on. One example of the schedule, say C_1, is illustrated in Fig. 2. In C_1, students s_1 through s_3, and students s_4 through s_6

⋆ Supported in part by Scientific Research Grant, Ministry of Japan, 16092223
⋆⋆ Supported in part by Scientific Research Grant, Ministry of Japan, 16092215 and 17700015

E. K. Burke, H. Rudová (Eds.): PATAT 2006, pp. 448–453. ISBN 80-210-3726-1.

are scheduled to rooms r_1 and r_2, respectively, in this order of timeslots. Then the cost of p_1 in the schedule C_1 is 2 since p_1 has to move twice, from r_1 to r_2 and then from r_2 to r_1. The costs of p_2 and p_3 are 1 and 0, respectively. As a result, the cost of the schedule C_1 is $2 + 1 + 0 + 1 + 0 + 0 = 4$.

Student	s_1	s_2	s_3	s_4	s_5	s_6
Assigned professors	p_1, p_2	p_2, p_3	p_1, p_4	p_4, p_5	p_1, p_6	p_2, p_6

Fig. 1. Example of referee-assignment

	r_1	r_2
t_1	s_1	s_4
t_2	s_2	s_5
t_3	s_3	s_6

C_1

	r_1	r_2
t_1	s_1	s_4
t_2	s_5	s_2
t_3	s_3	s_6

C_2

	r_1	r_2
t_1	s_3	s_6
t_2	s_1	s_4
t_3	s_2	s_5

C_3

Fig. 2. Schedules C_1, C_2 and C_3

It seems hard to attack this problem directly, and hence, we will consider two restricted problems, denoted ROOM and ORDER. ROOM takes an initial feasible solution as an input, as well as an assignment of referees to students. We are allowed only to exchange the presentation rooms of a pair of students assigned to the same timeslot; so it is prohibited to change the assigned timeslots. For example, C_2 in Fig. 2 is one possible output of ROOM when C_1 in Fig. 2 is an input schedule. C_2 is the result of exchanging rooms of s_2 and s_5, by which we can improve the cost of schedule to 3. The second problem, called ORDER, takes the same inputs as ROOM. It allows to exchange the timeslots of two pairs, but does not allow to change the assigned rooms. For example, C_3 in Fig. 2 is one possible output of ORDER when C_1 is an input. Recall that a feasible solution can be found in polynomial time as mentioned above, and hence, allowing to give an initial schedule as an input is reasonable.

Our Contribution. In this paper, we investigate the time complexity of ROOM and ORDER. It is reasonable to assume that the number of students each professor evaluates, and the number of professors assigned to each student are bounded by, say, s and t, respectively. Problem ROOM(s, t) is ROOM whose input is restricted as above, and problem ORDER(s, t) is defined similarly. By definition, ROOM$(1, t)$ is trivially in $\mathcal{P}$ for any t. This paper shows that (i) ROOM$(2, 1)$ is also polynomial-time solvable, but (ii) ROOM(s, t) is generally $\mathcal{NP}$-hard and furthermore it remains intractable even if $s = 2$ and $t = 2$. As for the complex-

ity of ORDER, it is easy to see that ORDER(s,t) is polynomial-time solvable for $s \leq 2$. We show that (iii) ORDER$(3,1)$ is in $\mathcal{P}$.

Related Work. In educational timetabling, a set of resources such as teachers, students, rooms, and lectures must be assigned to a set of timeslots subject to certain hard and soft constraints. There are three main categories in educational timetabling, namely, school (or class-teacher), university course, and exam timetabling (e.g., see [7]). There are a large number of researchers investigating in detail the complexity of university timetabling [2–6, 8].

The interview timetabling problem treated in this paper can be regarded as the classical examination timetabling problem by considering students and referees in the former problem as exams and students in the latter problem, respectively. However, interesting parameter setting where we can derive a boundary between $\mathcal{P}$and $\mathcal{NP}$-hard is the case when s and t are small. For such settings, it is natural to interpret the problem as the interview timetabling rather than examination timetabling.

2 Complexity of ROOM

It would be trivial that ROOM$(1,t)$ is in $\mathcal{P}$ for any t because no professor needs to move and hence the cost is 0 for any schedule. However, the precise complexity of ROOM(s,t) for $s \geq 2$ is not evident. First, we consider ROOM$(2,1)$ and give a polynomial-time algorithm that solves it.

Let us call two students who are assigned to the same timeslot by an input schedule *a student-pair* (or simply, *a pair*). In the following, if we write a student-pair as (s_i, s_j), it means that s_i and s_j are assigned to rooms r_1 and r_2, respectively, by the current schedule. By "*flip a student-pair* (s_i, s_j)", we mean to exchange the rooms of s_i and s_j, namely, we change (s_i, s_j) to (s_j, s_i). Without loss of generality, we assume that each student is evaluated by exactly one professor (namely, there is no student to whom no referee is assigned). So, if professor p is assigned to student s, we write $A(s) = p$ for convenience.

Starting from an initial schedule C, our algorithm decides, for each student-pair, whether to flip it or not, in a sequential manner. We first select an arbitrary pair, say (u, v), and fix the rooms of this pair as it is, and focus on the student u. We select a pair (x, y) (if any) such that either x or y is evaluated by $A(u)$. If it is x, namely $A(u) = A(x)$, we keep the rooms of x and y as it is, so that the professor $A(u)$ does not have to move. If it is y, then we flip the pair (x, y), again, so that $A(u)$ need not move. In this way, we continue determining the rooms of pairs, so that a professor in question does not have to move. When there is no pair to select, then we focus on the other student v of the initial pair, and do the same operation starting from v. If there is no pair to select, we close this "chain", and start the next phase by selecting an initial pair again. The algorithm stops when all pairs are processed.

Theorem 1. ROOM$(2,1)$ *is in* $\mathcal{P}$.

Proof (Sketch). Clearly, the time complexity is polynomial. It is not hard to see that each phase gives rise to the cost of at most one, and if it is one, then the set of pairs selected in that phase causes the cost of one in any schedule. Hence the schedule the above algorithm outputs is optimal. □

Next, we show an intractable case of ROOM(s,t).

Theorem 2. ROOM*(s,t) for $s \geq 2, t \geq 2$ is $\mathcal{NP}$-hard.*

Proof (Sketch). Consider the following problem MAX E2LIN2(3): We are given n variables $x_1, x_2, \cdots, x_n$ and m equations each with exactly two variables, $x_{i_1} \oplus x_{i_2} = a_i$ $(1 \leq i \leq m,\ a_i \in \{0,1\})$ such that each variable appears at most three times in the equations. We are asked to assign 0 or 1 to variables so that the number of satisfied equations is maximized. It is known that MAX E2LIN2(3) is $\mathcal{NP}$-hard [1].

Given an instance I of MAX E2LIN2(3) with n variables and m equations, we construct an instance I' of ROOM(2,2). For each variable x_i $(1 \leq i \leq n)$, we create a student-pair $(s_{i,1}, s_{i,2})$, and for each equation $e_j : x_{j_1} \oplus x_{j_2} = a_j$ $(1 \leq j \leq m)$, we create a professor p_j. Referee-assignment is constructed as follows. Consider the j-th equation e_j $(1 \leq j \leq m)$. If it is of the form $x_{j_1} \oplus x_{j_2} = 0$, then either `(a1)` assign p_j to $s_{j_1,1}$ and $s_{j_2,1}$, or `(a2)` assign p_j to $s_{j_1,2}$ and $s_{j_2,2}$. If $x_{j_1} \oplus x_{j_2} = 1$, then either `(b1)` assign p_j to $s_{j_1,1}$ and $s_{j_2,2}$, or `(b2)` assign p_j to $s_{j_1,2}$ and $s_{j_2,1}$. Observe that each professor appears twice in I' since each equation contains two variables, but three referees may be assigned to one student since a variable can appear three times. Hence, at this moment, a constructed instance is of ROOM$(2,3)$. However, we can create an instance of ROOM$(2,2)$ by appropriately choosing `(a1)` or `(a2)` (`(b1)` or `(b2)`), although we omit describing how to do it.

An assignment C for I naturally corresponds to a schedule C' of I': If $x_i = 0$, then the rooms of $(s_{i,1}, s_{i,2})$ is the same as in the initial schedule. If $x_i = 1$, the rooms of $(s_{i,1}, s_{i,2})$ is flipped to $(s_{i,2}, s_{i,1})$. It is not hard to see that the number of unsatisfied equations under C is equal to the total number of movements of professors under C'. □

3 Complexity of ORDER

Recall that the operation we are allowed in this problem is only to decide the timeslot of student-pairs. Hence, as the simplest example, if each professor judges at most two students, any exchange operation of timeslots of two student-pairs does not change the cost of the schedule. It follows that ORDER(s,t) is in $\mathcal{P}$ for $s \leq 2$ and any t since any solution is optimal.

In this section we present a polynomial-time algorithm to find an optimal solution for ORDER$(3,1)$. Let $cost(C,p)$ be the number of movements of professor p under a schedule C. Note that if a professor p appears at most twice in an input referee-assignment, $cost(C,p)$ is the same for any schedule C as mentioned above. Even if p appears three times, $cost(C,p) = 0$ for any C if all three students

are assigned to the same room by the input schedule. These professors are called *non-potential* professors. If p appears three times, and if two of his/her students are assigned to the same room, and the other one is assigned to the other room, his/her cost can be one or two depending on the schedule. We call these professors *potential* professors. As in the case of ROOM$(2,1)$, let $A(s)$ denote the referee assigned to student s.

As before, we sequentially determine the schedule of each pair. We construct several blocks of student-pairs. Starting from an arbitrary initial student-pair (u,v), we select a student-pair (x,y) where $A(u)=A(x)$ and $A(u)$ is a potential professor, if any. We schedule (x,y) to the timeslot next to (u,v), so that two students professor $A(u)$ evaluates are assigned to continuous timeslots and to the same room. Next, we select a pair (w,z) such that $A(y)=A(z)$ and $A(y)$ is a potential professor, if any, and schedule (w,z) to the timeslot next to (x,y), and so on. When there is no pair to select, we then go back to (u,v), and perform the same operation starting from $A(v)$. This time, we schedule new pairs to *previous* timeslots of (u,v). When there is no pair to select, the work on the current block is finished, and we start to construct a new block by selecting an arbitrary initial student-pair. Finally, blocks are scheduled in an arbitrary order.

Theorem 3. ORDER$(3,1)$ *is in* $\mathcal{P}$. *(Proof is omitted.)*

4 Concluding Remarks

In this paper, we considered the time complexity of ROOM(s,t) and ORDER(s,t) for several values of s and t. The apparent next step in this research is to investigate the complexity of ROOM$(3,1)$ and ORDER$(4,1)$. An interesting generalization is to allow both operations of ROOM and ORDER simultaneously. The goal in this line is to consider the most general problem, namely, the problem without an initial schedule in an input.

References

1. P. Berman and M. Karpinski, "Improved approximation lower bounds on small occurrence optimization," ECCC Report, TR03-008, 2003.
2. M. W. Carter and C. A. Tovey, "When is the classroom assignment problem hard?" *Operations Research*, Vol.40, No.1, pp.28–39, 1992.
3. E. Cheng, S. Kruk, and M. J. Lipman, "Flow formulations for the student scheduling problem," in *Proc. 4th PATAT 2002*, LNCS 2740, pp.299–309, 2003.
4. T. B. Cooper, J. H. Kingston, "The complexity of timetable construction problems," *Proc. 1st PATAT 1995*, LNCS 1153, pp. 283–295, 1996.
5. H. M. M. ten Eikelder, and R.J. Willemen, "Some complexity aspects of secondary school timetabling problems," in *Proc. 3rd PATAT 2000*, LNCS 2079, pp.18–27, 2001.
6. S. Even, A. Itai, and A. Shamir, " On the complexity of timetabling and multicommodity flow problems," *SIAM J. Computing*, Vol.5, No.4, pp.691–703, 1976.

7. A. Schaerf, "A survey of automated timetabling," *Artificial Intelligence Review*, Vol.13, No.2, pp.87–127, 1999.
8. D. de Werra, "Some Combinatorial Models for Course Scheduling," *Proc. 1st PATAT 1995*, LNCS 1153, pp.296–308, 1996.