Discrete Applied Mathematics 308 (2022) 220–234

Contents lists available at ScienceDirect

Discrete Applied Mathematics

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

Student-project allocation with preferences over projects: Algorithmic and experimental results[☆]

David Manlove[1], Duncan Milne, Sofiat Olaosebikan[*,2]

School of Computing Science, University of Glasgow, United Kingdom of Great Britain and Northern Ireland

ARTICLE INFO

Article history:
Received 30 November 2018
Received in revised form 9 August 2020
Accepted 10 August 2020
Available online 29 August 2020

Keywords:
Student-project allocation
Blocking pair
Coalition
Stable matching
Polynomial-time algorithm
Inapproximability
Integer programming
Empirical analysis

ABSTRACT

We study the Student-Project Allocation problem with lecturer preferences over Projects (SPA-P). In this context it is known that stable matchings can have different sizes and the problem of finding a maximum size stable matching is NP-hard. There are two known approximation algorithms for MAX-SPA-P, with performance guarantees 2 and $\frac{3}{2}$. We show that MAX-SPA-P is polynomial-time solvable if there is only one lecturer involved, and NP-hard to approximate within some constant $c > 1$ if there are two lecturers involved. We also show that this problem remains NP-hard if each preference list is of length at most 3, with an arbitrary number of lecturers. We then describe an Integer Programming (IP) model to enable MAX-SPA-P to be solved optimally in the general case. Following this, we present results arising from an empirical evaluation that investigates how the solutions produced by the approximation algorithms compare to optimal solutions obtained from the IP model, with respect to the size of the stable matchings constructed, on instances that are both randomly-generated and derived from real datasets.

1. Introduction

Matching problems, which generally involve the assignment of a set of agents to another set of agents based on preferences, have wide applications in many real-world settings. One such application can be seen in an educational context, e.g., the allocation of pupils to schools, school-leavers to universities and students to projects. In the context of allocating students to projects, university lecturers propose a range of projects, and each student is required to provide a preference over the available projects that she finds acceptable. Lecturers may also have preferences over the students that find their project acceptable and/or the projects that they offer. There may also be upper bounds on the number of students that can be assigned to a particular project, and the number of students that a given lecturer is willing to supervise. The problem then is to allocate students to projects based on these preferences and capacity constraints — the so-called *Student-Project Allocation problem* (SPA) [5,15].

[☆] A preliminary version of a part of this paper appeared in Manlove et al. (2018).

[*] Corresponding author.

E-mail addresses: David.Manlove@glasgow.ac.uk (D. Manlove), Duncan.Milne1@gmail.com (D. Milne), Sofiat.Olaosebikan@glasgow.ac.uk (S. Olaosebikan).

URL: http://www.sofiatolaosebikan.com (S. Olaosebikan).

[1] Supported by grant EP/P028306/1 from the Engineering and Physical Sciences Research Council, United Kingdom.

[2] Supported by a College of Science and Engineering Scholarship, University of Glasgow, United Kingdom.

https://doi.org/10.1016/j.dam.2020.08.015

Two major models of SPA exist in the literature: one permits preferences only from the students [2,10,14,18], while the other permits preferences from the students and lecturers [1,12]. Given the large number of students that are typically involved in such an allocation process, many university departments seek to automate the allocation of students to projects. Examples include the School of Computing Science, University of Glasgow [14], the Faculty of Science, University of Southern Denmark [5], the Department of Computing Science, University of York [12], and elsewhere [2,4,10,20].

In general we seek a *matching*, which is a set of agent pairs who find one another acceptable that satisfies the capacities of the agents involved. For matching problems where preferences exist from the two sets of agents involved (e.g., junior doctors and hospitals in the classical *Hospitals–Residents problem* (HR) [7], or students and lecturers in the context of SPA), it has been argued that the desired property for a matching one should seek is that of *stability* [19]. Informally, a *stable matching* ensures that no acceptable pair of agents who are not matched together would rather be assigned to each other than remain with their current assignees.

Abraham et al. [1] described two linear-time algorithms to find a stable matching in a variant of SPA where students have preferences over projects whilst lecturers have preferences over students. The first algorithm finds the *student-optimal* stable matching, in the sense that each assigned student is allocated to the best project that she could obtain in any stable matching; while the second algorithm outputs the *lecturer-optimal* stable matching, in the sense that each assigned lecturer is allocated the best set of students that she could obtain in any stable matching.

Manlove and O'Malley [17] proposed another variant of SPA where both students and lecturers have preferences over projects, referred to as SPA-P. A motivation for this variant is that a lecturer might prefer to supervise projects that are closely related to or in-line with her research interests, while the remaining projects that she proposed, perhaps to ensure that the students have some sense of choice, are of lesser priority to her. In their paper, they formulated an appropriate stability definition for SPA-P, and they showed that stable matchings in this context can have different sizes. Moreover, in addition to stability, a very important requirement in practice is to match as many students to projects as possible.

Consequently, Manlove and O'Malley [17] proved that MAX-SPA-P, the problem of finding a maximum size stable matching given an instance of SPA-P, is NP-hard. Further, they showed that MAX-SPA-P is not approximable within δ_1, for some $\delta_1 > 1$, unless P = NP. Moreover, the result holds even if each project and lecturer has capacity 1, and all preference lists are of length at most 4. In addition, they gave a polynomial-time 2-approximation algorithm for MAX-SPA-P.

Subsequently, Iwama et al. [11] described an improved approximation algorithm with an upper bound of $\frac{3}{2}$, which builds on Manlove and O'Malley's algorithm [17] and uses ideas from Király's approximation algorithm for the problem of finding a maximum size stable matching in a variant of the *Stable Marriage problem* [13]. In addition, Iwama et al. [11] showed that MAX-SPA-P is not approximable within $\frac{21}{19} - \varepsilon$, for any $\varepsilon > 0$, unless P = NP.

Considering the fact that the existing algorithms for MAX-SPA-P are only guaranteed to produce an approximate solution, we seek another technique to enable MAX-SPA-P to be solved optimally. Integer Programming (IP) is a powerful technique for producing optimal solutions to a range of NP-hard optimisation problems, with the aid of commercial optimisation solvers, e.g., Gurobi [9], GLPK [8] and CPLEX [6]. These solvers can allow IP models to be solved in a reasonable amount of time, even with respect to problem instances that occur in practical applications.

Our contribution. We present new algorithmic and experimental results for SPA-P. On the algorithmic side, first we explore the complexity of MAX-SPA-P under the following separate restrictions: if (i) the instance involves r lecturers, denoted MAX-SPA-P-Lr, where r is a constant; and (ii) each preference list is of length at most 3, denoted (3, 3)-MAX-SPA-P. For the first restriction, if there is only one lecturer involved, we show that MAX-SPA-P is polynomial-time solvable. In contrast to this, if there are two lecturers involved, we show that the problem remains NP-hard and is not approximable within some constant $c > 1$ unless P = NP. For the second restriction, we show that MAX-SPA-P remains NP-hard. We then move on to describe an IP model to enable MAX-SPA-P to be solved optimally, in the general case where there are no restrictions on the problem instance. We remark that this IP model, excluding its correctness result, first appeared in [16].

On the experimental side, we present results arising from an empirical evaluation that investigates how the solutions produced by the existing approximation algorithms for MAX-SPA-P [11,17] compare to optimal solutions obtained from our IP model, with respect to the size of the stable matchings constructed, on instances that are both randomly-generated and derived from real datasets. These real datasets are based on actual student preference data and manufactured lecturer preference data from previous runs of student-project allocation processes at the School of Computing Science, University of Glasgow. We also present results showing the time taken by the IP model to solve the problem instances optimally. Our main finding is that the $\frac{3}{2}$-approximation algorithm finds stable matchings that are very close to having maximum cardinality over the tested instances.

The remainder of this paper is organised as follows. We give preliminary definitions and results in Section 2. In Section 3, we focus on MAX-SPA-P-Lr, with $r \in \{1, 2\}$; we describe a polynomial-time algorithm for MAX-SPA-P-L1 in Section 3.1, and we give an inapproximability result for MAX-SPA-P-L2 in Section 3.2. In Section 4 we show that (3, 3)-MAX-SPA-P is NP-hard. In Section 5, we describe our IP model for MAX-SPA-P. We present our empirical evaluation in Section 6, along with some discussions regarding results obtained from the experiments. Finally, in Section 7 we give some conclusions and open problems.

Students' preferences	Lecturers' preferences
s_1: p_3 p_2 p_1	l_1: p_2 p_1
s_2: p_1 p_2	l_2: p_3
s_3: p_3	
	Project capacities: $c_1 = c_2 = c_3 = 1$
	Lecturer capacities: $d_1 = 2,\ d_2 = 1$

Fig. 1. An instance I_1 of SPA-P.

2. Preliminary definitions and results

We give a formal definition for SPA-P as described in the literature [17]. An instance I of SPA-P involves a set $\mathcal{S} = \{s_1, s_2, \ldots, s_{n_1}\}$ of *students*, a set $\mathcal{P} = \{p_1, p_2, \ldots, p_{n_2}\}$ of *projects* and a set $\mathcal{L} = \{l_1, l_2, \ldots, l_{n_3}\}$ of *lecturers*. Each lecturer $l_k \in \mathcal{L}$ offers a non-empty subset of projects, denoted by P_k. We assume that $P_1, P_2, \ldots, P_{n_3}$ partitions $\mathcal{P}$ (i.e., each project is offered by one lecturer). Also, each student $s_i \in \mathcal{S}$ has an *acceptable* set of projects $A_i \subseteq \mathcal{P}$. We call a pair $(s_i, p_j) \in \mathcal{S} \times \mathcal{P}$ an *acceptable pair* if $p_j \in A_i$. Moreover s_i ranks A_i in strict order of preference. Similarly, each lecturer l_k ranks P_k in strict order of preference. Finally, each project $p_j \in \mathcal{P}$ and lecturer $l_k \in \mathcal{L}$ has a positive capacity denoted by c_j and d_k respectively.

An *assignment* M is a subset of $\mathcal{S} \times \mathcal{P}$ where $(s_i, p_j) \in M$ implies that s_i finds p_j acceptable (i.e., $p_j \in A_i$). We define the *size* of M as the number of (student, project) pairs in M, denoted $|M|$. If $(s_i, p_j) \in M$, we say that s_i is *assigned to* p_j and p_j is *assigned* s_i. Furthermore, we denote the set of projects assigned to student s_i in M as $M(s_i)$. Similarly, we denote the set of students assigned to project p_j in M as $M(p_j)$. For ease of exposition, if s_i is assigned to a project p_j offered by lecturer l_k, we may also say that s_i is *assigned to* l_k, and l_k is *assigned* s_i. Thus we denote the set of students assigned to l_k in M as $M(l_k)$.

A project $p_j \in \mathcal{P}$ is *full*, *undersubscribed* or *oversubscribed* in M if $|M(p_j)|$ is equal to, less than or greater than c_j, respectively. The corresponding terms apply to each lecturer l_k with respect to d_k. We say that a project $p_j \in \mathcal{P}$ is *non-empty* if $|M(p_j)| > 0$. Similarly we say that a lecturer $l_k \in \mathcal{L}$ is *non-empty* if $|M(l_k)| > 0$.

A *matching* M is an assignment such that $|M(s_i)| \leq 1$ for each $s_i \in \mathcal{S}$, $|M(p_j)| \leq c_j$ for each $p_j \in \mathcal{P}$, and $|M(l_k)| \leq d_k$ for each $l_k \in \mathcal{L}$ (i.e., each student is assigned to at most one project, and no project or lecturer is oversubscribed). In what follows, l_k is the lecturer who offers p_j. Given a matching M, an acceptable pair $(s_i, p_j) \in (\mathcal{S} \times \mathcal{P}) \setminus M$ is a *blocking pair* for M if (1) and (2) holds as follows:

(1) either s_i is unassigned in M or s_i prefers p_j to $M(s_i)$;

(2) p_j is undersubscribed, and either (a), (b), or (c) holds as follows:

(a) $s_i \in M(l_k)$ and l_k prefers p_j to $M(s_i)$;
(b) $s_i \notin M(l_k)$ and l_k is undersubscribed;
(c) $s_i \notin M(l_k)$ and l_k prefers p_j to her worst non-empty project.

If such a pair were to occur in practice, s_i and l_k will have an incentive to form a private arrangement outside the matching in order to improve on their current allocation via p_j (see [17, Section 2] for an intuition for this definition). Henceforth, if (s_i, p_j) forms a blocking pair for M then we refer to (s_i, p_j) as a blocking pair of type (a), type (b), or type (c), according as (s_i, p_j) satisfies conditions 2(a), 2(b), or 2(c), respectively.

With respect to the SPA-P instance given in Fig. 1, $M_1 = \{(s_1, p_3), (s_2, p_1)\}$ is clearly a matching. It is obvious that each of students s_1 and s_2 is matched to her first ranked project in M_1. Although s_3 is unassigned in M_1, the lecturer offering p_3 (the only project that s_3 finds acceptable) is assumed to be indifferent among those students who find p_3 acceptable. Also p_3 is full in M_1. Thus, we say that M_1 admits no blocking pair.

Another way in which a matching could be undermined is through a group of students acting together. Given a matching M, a *coalition* is a set of students $C = \langle s_{i_0}, \ldots, s_{i_{r-1}} \rangle$, for some $r \geq 2$ such that each student s_{i_j} $(0 \leq j \leq r-1)$ is assigned in M and prefers $M(s_{i_{j+1}})$ to $M(s_{i_j})$, where addition is performed modulo r. With respect to Fig. 1, the matching $M_2 = \{(s_1, p_1), (s_2, p_2), (s_3, p_3)\}$ admits a coalition $\{s_1, s_2\}$, as students s_1 and s_2 would rather permute their assigned projects in M_2 so as to be better off. We note that the number of students assigned to each project and lecturer involved in any such swap remains the same after such a permutation. Moreover, the lecturers involved would have no incentive to prevent the switch from occurring since they are assumed to be indifferent between the students assigned to the projects they are offering. If a matching admits no coalition, we define such matching to be *coalition-free*.

Given an instance I of SPA-P, we define a matching M in I to be *stable* if M admits no blocking pair and is coalition-free. It turns out that with respect to this definition, stable matchings in I can have different sizes. Clearly, each of the matchings $M_1 = \{(s_1, p_3), (s_2, p_1)\}$ and $M_3 = \{(s_1, p_2), (s_2, p_1), (s_3, p_3)\}$ is stable in the SPA-P instance I_1 shown in Fig. 1. The varying sizes of the stable matchings produced naturally leads to the problem of finding a maximum cardinality stable matching given an instance of SPA-P, which we denote by MAX-SPA-P.

Algorithm 1 Algorithm MAX-SPA-P-L1

```
Input: an instance $I$ of SPA-P-L1
Output: a maximum size stable matching in $I$
$M \leftarrow$ a greedy maximum matching in $I$
while there exist a type (c) blocking pair do
    $s_i \leftarrow$ a student involved in such blocking pair
    $(s_i, p_j) \leftarrow$ best blocking pair for $s_i$
    $p_q \leftarrow$ worst non-empty project offered by $l$
    $s_r \leftarrow$ any student assigned to $p_q$ in $M$
    $M \leftarrow M \setminus \{(s_r, p_q)\}$
    $M \leftarrow M \cup \{(s_i, p_j)\}$ /* promote $s_i$ to $p_j$ */
while $M$ admits a coalition $C$ do
    satisfy $C$
return $M$
```

3. SPA-P with constant number of lecturers

As mentioned in Section 1, MAX-SPA-P is not approximable within δ_1, for some $\delta_1 > 1$ unless P = NP [17]. Moreover, the result holds even if each project and lecturer has capacity 1, and each preference list is of length at most 4. We wished to answer the following question: what is the complexity of finding a maximum size stable matching if a constant number of lecturers are involved in an arbitrary SPA-P instance I? As it turns out, the problem is polynomial-time solvable if there is only one lecturer involved in I and hard to approximate if there are two lecturers involved in I. We present the proof of these results in the next two sections.

We denote by SPA-P-Lr an instance of SPA-P involving r lecturers, where r is a constant, and we denote by MAX-SPA-P-Lr the problem of finding a maximum size stable matching given an instance of SPA-P-Lr.

3.1. MAX-SPA-P-L1 is polynomial-time solvable

3.1.1. Introduction

Let I be an instance of SPA-P-L1 where l is the lecturer involved in I; assume all notation and terminology from the general SPA-P case. Given an acceptable pair (s_i, p_j), we define rank(s_i, p_j) as 1 plus the number of projects that s_i prefers to p_j. Let R be the maximum rank of a project in any student's preference list. We remark that each of rank(s_i, p_j) and R is calculated from the problem instance.

We define the *profile* of a matching M in I, $\rho(M)$, as an R-tuple $(x_1, x_2, \ldots, x_R)$, such that for each r $(1 \leq r \leq R)$, x_r is the number of students, say s_i, assigned in M to a project, say p_j, such that rank$(s_i, p_j) = r$. For example, the matching $M_3 = \{(s_1, p_2), (s_2, p_1), (s_3, p_3)\}$ in the instance I_1 given in Fig. 1 has the profile $\rho(M_3) = (2, 1, 0)$, since two students (i.e., s_2 and s_3) are assigned to their first choice project, one student (i.e., s_1) is assigned to her second choice project, and no student is assigned to her third choice project. A *greedy maximum matching* in I is a matching of maximum cardinality that has lexicographically maximum profile. Clearly, M_3 is a greedy maximum matching in I_1.

3.1.2. Description of the algorithm

Our algorithm, which we denote by Algorithm MAX-SPA-P-L1, begins by first constructing a greedy maximum matching M in I using the polynomial-time algorithm given in [14]. As we will prove later, at this point in the algorithm, M is coalition-free and does not admit a blocking pair of type (a) or type (b). If M admits a type (c) blocking pair then the first while loop of the algorithm is executed. In this loop we identify a student, say s_i, involved in a type (c) blocking pair, say (s_i, p_j), such that (s_i, p_j) is the best blocking pair for s_i. Next, we identify l's worst non-empty project in M, say p_q; we identify any student assigned to p_q in M, say s_r, and we remove the pair (s_r, p_q) from M. Finally, we promote s_i by adding the pair (s_i, p_j) to M. As we will prove later, M does not admit a blocking pair at the termination of this loop.

If M admits a coalition $C = \langle s_{i_0}, s_{i_1}, \ldots, s_{i_{r-1}} \rangle$ at this point, for some $r \geq 2$, then we *satisfy* the coalition in the second while loop, by letting the students in C swap their projects, i.e., for each j $(0 \leq j \leq r-1)$, s_{i_j} moves project from $M(s_{i_j})$ to $M(s_{i_{j+1}})$, where addition is taken modulo r. At the termination of this loop, M is output as a maximum size stable matching in I. We describe our algorithm in pseudocode form in Algorithm 1.

3.1.3. Correctness of the algorithm

We present the following theorem concerning the correctness of our algorithm.

Theorem 1. *Given an instance I of SPA-P-L1, Algorithm MAX-SPA-P-L1 terminates with a maximum size stable matching in $O(n_1^2 Rm)$ time, where n_1 is the number of students, R is the maximum rank of a project in any student's preference list and m is the total length of the students' preference lists.*

Proof. Let E be an arbitrary execution of Algorithm MAX-SPA-P-L1, and let M be the matching at the termination of E. Let M_0 be the greedy maximum matching in I found at line 1 during E; clearly M_0 is of maximum cardinality. In what follows, we show that just before the first while loop was initiated during E, M_0 admits no blocking pair of type (a) or type (b), and is coalition-free.

Suppose that (s_i, p_j) is a blocking pair of type (a), i.e., $s_i \in M_0(l)$ and s_i prefers p_j to $M_0(s_i)$; p_j is undersubscribed in M_0 and l prefers p_j to $M_0(s_i)$. Let $M_0(s_i) = p_{j'}$. Now, let $M_0' = (M_0 \setminus \{(s_i, p_{j'})\}) \cup \{(s_i, p_j)\}$. Then $|M_0'| = |M_0|$ and $\rho(M_0') > \rho(M_0)$ (according to lexicographic order), contradicting the fact that M_0 is a greedy maximum matching. Suppose that (s_i, p_j) is a blocking pair of type (b), i.e., s_i is unassigned in M_0 and each of p_j and l is undersubscribed in M_0. Now, p_j undersubscribed and l undersubscribed implies that $M_0 \cup \{(s_i, p_j)\}$ is a matching in I, contradicting the maximality of M_0. Next, we show that M_0 is coalition-free. Suppose M_0 admits a coalition of students $C = \langle s_{i_0}, s_{i_1}, \dots, s_{i_{r-1}} \rangle$ for some $r \geq 2$. Suppose M_0' is the matching that results from satisfying C. Then $|M_0'| = |M_0|$ and $\rho(M_0') > \rho(M_0)$, since each student in C will prefer her assigned project in M_0' to her assigned project in M_0. This is a contradiction to the fact that M_0 has a lexicographically maximum profile.

We note that the first while loop is bound to terminate since (i) the number of students involved in I is finite, and (ii) for each student involved in a type (c) blocking pair that gets promoted, the lecturer obtains one more student assigned to a project that is better than her previous worst non-empty project. Let M_1 be the matching at the termination of the first loop. Clearly, M_1 cannot admit a type (c) blocking pair. Moreover, M_1 cannot admit a type (b) blocking pair either, since for each student that becomes unassigned from a project within the loop, another student becomes promoted to a project in the same while loop iteration. Thus the cardinality of M_1 remains unchanged, and the proof follows as in the previous paragraph.

Next, we show that M_1 cannot admit a blocking pair of type (a). Suppose that (s_i, p_j) is a blocking pair of type (a). This implies that $s_i \in M_1(l)$, p_j is undersubscribed in M_1, and l prefers p_j to $M_1(s_i)$. Let $M_1(s_i) = p_k$. First, we note that (s_i, p_k) must have been added to M_1 as a result of a type (c) blocking pair being satisfied within the first while loop. For, suppose otherwise, we have already established that M_0 admits no blocking pair of type (a). Hence p_j must have been full in M_0 and became undersubscribed during the first while loop. This is only possible if l's worst non-empty project in M_1 is p_j or better. Hence (s_i, p_j) cannot be a type (a) blocking pair for M_1 in this case. Thus (s_i, p_k) must have been added to M_1 within the first while loop.

Moreover, when (s_i, p_k) was added to the matching during the first while loop, p_j must be full, since we satisfy the best blocking pair for each student within a loop iteration. However, for (s_i, p_j) to form a type (a) blocking pair for M_1, p_j must be undersubscribed in M_1. Now, the only way p_j can end up becoming undersubscribed in M_1, having been full in M_0, is if p_j was l's worst non-empty project at a point where a type (c) blocking pair was identified during the first while loop. Thus in M_1, either p_j is l's worst non-empty project, or l's worst non-empty project is better than p_j. Hence (s_i, p_j) cannot be a type (a) blocking pair for M_1.

We have proved in the last two paragraphs that the first while loop terminates with a maximum matching M_1 in I, which admits no blocking pair. The second while loop is also bound to terminate, since (i) each student has a finite number of projects in their preference list, and (ii) for any coalition C of students that exists, each student involved in C obtains a better project after the swap takes place. Clearly M cannot admit any new blocking pair at the end of the second while loop. Hence, at the termination of the algorithm, M is a maximum size stable matching in I.

Analysis of the algorithm. Algorithm MAX-SPA-P-L1 finds a maximum size stable matching in I in three phases. In the first phase, the algorithm finds a greedy maximum matching M in I. This can be obtained in $O(n_1^2 Rm)$ time using the GREEDY-MAX-SPA algorithm described in [14], and this complexity dominates the overall runtime of the algorithm. In the second phase, the algorithm eliminates potential type (c) blocking pairs in M. This phase is bounded by the total length of the students' preference lists. Thus the complexity is $O(m)$.

In the third phase, the algorithm eliminates potential coalitions that might have been introduced in the second phase. We remark that coalitions in M correspond to cycles in the *envy graph* $G(M)$, which contains a vertex for each student who is assigned in M and a directed edge from student s_{i_j} to s_{i_q} if s_{i_j} prefers $M(s_{i_q})$ to $M(s_{i_j})$. Clearly M is coalition-free if and only if $G(M)$ is acyclic (see Section 5.1 for further discussions on this). Further, we remark that cyclic coalitions can be eliminated in $O(m)$ time (see [15, p. 308] for a detailed explanation on how to achieve this). Hence, the overall complexity of Algorithm MAX-SPA-P-L1 is $O(n_1^2 Rm)$.

3.2. MAX-SPA-P-L2 *is hard to approximate*

We define MAX-SPA-P-L2 to be the problem of finding a maximum size stable matching, given an instance of SPA-P-L2. In this section we show that it is NP-hard to approximate MAX-SPA-P-L2 within a factor of δ, for some $\delta > 1$. First, we define a problem relating to matchings in graphs. A matching M in a graph G is said to be *maximal* if no proper superset of M is a matching in G.

Define MIN-MM to be the problem of computing the minimum size of a maximal matching, given a graph G (where G is a subdivision graph of some cubic graph G'). Then $G = (U, W, E)$ is a bipartite graph, where, without loss of generality, each vertex in U has degree 2 and each vertex in W has degree 3. Let $U = \{u_1, u_2, \dots, u_{n_1}\}$ and $W = \{w_1, w_2, \dots, w_{n_2}\}$. For each $u_i \in U$, let w_{j_i} and w_{k_i} be the two neighbours of u_i in G, where $j_i < k_i$. Manlove and O'Malley [17] showed that

Students' preferences		Lecturers' preferences	
u_i^1: $r_i\ p_{j_i}\ p_{k_i}\ t_i$	$(1 \leq i \leq n_1)$	w_j: $p_j\ q_j$	$(1 \leq j \leq n_2)$
u_i^2: $r_i\ p_{k_i}\ p_{j_i}$	$(1 \leq i \leq n_1)$	x_j: r_j	$(1 \leq j \leq n_1)$
s_i: q_i	$(1 \leq i \leq n_2)$	y_j: t_j	$(1 \leq j \leq n_1)$

Fig. 2. Preference lists for constructed instance of SPA-P [17, Section 3]. Each project and lecturer has capacity 1.

Students' preferences		Lecturers' preferences
u_i^1: $r_i\ p_{j_i}\ p_{k_i}\ t_i$	$(1 \leq i \leq n_1)$	w: $p_1 \cdots p_{n_2}\ q_1 \ldots q_{n_2}$
u_i^2: $r_i\ p_{k_i}\ p_{j_i}$	$(1 \leq i \leq n_1)$	x: $r_1 \ldots r_{n_1}\ t_1 \ldots t_{n_1}$
s_i': q_i	$(1 \leq i \leq n_2)$	

Fig. 3. Preference lists for constructed instance of SPA-P-L2. Each project has capacity 1, whilst lecturers w and x has capacity n_2 and $2n_1$ respectively.

it is NP-hard to approximate MAX-SPA-P within δ, for some constant $\delta > 1$, using a reduction from MIN-MM, creating a SPA-P instance I as shown in Fig. 2.

The instance in Fig. 2 consists of $2n_1+n_2$ students in the set $S \cup U^1 \cup U^2$, where $S = \{s_1, s_2, \ldots, s_{n_2}\}$, $U^1 = \{u_i^1 : u_i \in U\}$ and $U^2 = \{u_i^2 : u_i \in U\}$; $2n_2 + 2n_1$ projects in the set $P \cup Q \cup R \cup T$, where $P = \{p_1, p_2, \ldots, p_{n_2}\}$, $Q = \{q_1, q_2, \ldots, q_{n_2}\}$, $R = \{r_1, r_2, \ldots, r_{n_1}\}$ and $T = \{t_1, t_2, \ldots, t_{n_1}\}$; and $2n_1+n_2$ lecturers in the set $W \cup X \cup Y$, where $X = \{x_1, x_2, \ldots, x_{n_1}\}$ and $Y = \{y_1, y_2, \ldots, y_{n_1}\}$. In addition, each project and lecturer has capacity 1; thus, the total lecturer capacity is $2n_1+n_2$. We will show that their reduction holds even if instead we create a SPA-P-L2 instance I' involving two lecturers with a total lecturer capacity of $2n_1+n_2$, where the number of students and projects remains the same. We illustrate our constructed SPA-P-L2 instance in Fig. 3, which consists of the set of students $S' \cup U^1 \cup U^2$, where $S' = \{s_1', s_2', \ldots, s_{n_2}'\}$, the set of projects $P \cup Q \cup R \cup T$ and the set of lecturers $\{w, x\}$. Each project has capacity 1, whilst lecturers w and x have capacity n_2 and $2n_1$ respectively.

Let I and I' be the SPA-P and SPA-P-L2 instances as defined above, respectively. The next two propositions demonstrate that the size of a maximum stable matching in I is equal to that in I'.

Proposition 2. *A stable matching M in I corresponds to a stable matching M' in I', where $|M| = |M'|$.*

Proof. Let M be a stable matching in I. We construct a matching M' in I' from M in the following way. First, all the pairs in M involving students in $U^1 \cup U^2$ are added to M'. Next, if k students in S are assigned to a project in M, we add (s_z', q_z) to M' for $1 \leq z \leq k$.

Now, we show that M' is a stable matching in I'. Suppose (u_i^z, r_i) is a blocking pair of M' in I', for some $i\ (1 \leq i \leq n_1)$ and some $z\,(1 \leq z \leq 2)$. This implies that u_i^z is either unassigned in M' or prefers r_i to her assigned project in M'. By the construction of M' from M, the same is true for u_i^z in M. Moreover, we have that r_i is undersubscribed in M', and thus r_i is undersubscribed in M. This implies that the lecturer who offers r_i in I, i.e., x_i, is undersubscribed in M. Hence (u_i^z, r_i) is a blocking pair of M in I, a contradiction.

Next, suppose (u_i^z, p_j) is a blocking pair of M' in I', for some $i\,(1 \leq i \leq n_1)$, some $j\,(1 \leq j \leq n_2)$, and some $z\,(1 \leq z \leq 2)$. This implies that p_j is undersubscribed in M', and thus p_j is undersubscribed in M. Moreover, we have that the lecturer who offers p_j in I, i.e., w_j, is either undersubscribed in M or is full in M with its sole student assigned to q_j. Further, u_i^z is either unassigned in M' or prefers p_j to her assigned project in M', and the same is true for u_i^z in M. Hence (u_i^z, p_j) is a blocking pair of M in I, a contradiction. Thus no project in P can be involved in a blocking pair of M' in I'. Moreover, no project in R can be involved in a blocking pair of M' in I', since each such project is full in M'. Hence, no student in U^2 can be involved in a blocking pair of M' in I'.

Next, suppose (u_i^1, t_i) is a blocking pair of M' in I', for some $i\ (1 \leq i \leq n_1)$. This implies that u_i^1 is unassigned in M', and thus u_i^1 is unassigned in M. Moreover, t_i is undersubscribed in M' and thus t_i is undersubscribed in M. It follows that the lecturer who offers t_i in I, i.e., y_i, is undersubscribed in M. Hence (u_i^1, t_i) is a blocking pair of M in I, a contradiction. Hence, no student in U^1 can be involved in a blocking pair of M' in I'.

Next, suppose (s_i', q_i) is a blocking pair of M' in I', for some $i\ (1 \leq i \leq n_2)$. This implies that s_i' is unassigned in M'. By the construction of M' from M, we have that each of q_i and w is undersubscribed in M'. Now since the capacity of w is n_2, there exists some w_j, for some $j\,(1 \leq j \leq n_2)$, such that w_j is undersubscribed in M. Hence (s_j, q_j) is a blocking pair of M in I, a contradiction. Hence, no student in S' can be involved in a blocking pair of M' in I'.

We next verify that M' is coalition-free in I'. Clearly, no student in S' can be involved in a coalition, since any such student who is assigned in M' has her first choice project. Also, no student who is assigned in M' to a project in R can be in a coalition, since every such student is assigned to her first choice project. As a consequence, no student who is assigned in M' to her second choice project can be in a coalition, since each such student prefers only a project in R. Finally, no student in U^1 who is assigned to a project in T can be in a coalition, since no assigned student prefers a project in T to her project in M'. Hence M' is stable in I'. It is clear that $|M| = |M'|$. □

Proposition 3. *A stable matching M' in I' corresponds to a stable matching M in I, where $|M'| = |M|$.*

Proof. Let M' be a stable matching in I'. We construct a matching M in I from M' in the following way. First, all the pairs in M' involving students in $U^1 \cup U^2$ are added to M. Next, for each $j\,(1 \le j \le n_2)$ such that p_j is undersubscribed in M', we add (s_j, q_j) to M.

Now, we show that M is a stable matching in I. Suppose (u_i^z, r_i) is a blocking pair of M in I, for some $i\,(1 \le i \le n_1)$ and some $z\,(1 \le z \le 2)$. This implies that u_i^z is either unassigned in M or prefers r_i to her assigned project in M. By the construction of M from M', the same is true for u_i^z in M'. Moreover, we have that r_i is undersubscribed in M, and thus r_i is undersubscribed in M'. This implies that the lecturer who offers r_i in I' (i.e., x) is undersubscribed in M' (since the capacity of x is $2n_1$). Hence (u_i^z, r_i) is a blocking pair of M' in I', a contradiction.

Next, suppose (u_i^z, p_j) is a blocking pair of M in I, for some $i\,(1 \le i \le n_1)$, some $j\,(1 \le j \le n_2)$, and some $z\,(1 \le z \le 2)$. This implies that p_j is undersubscribed in M and thus p_j is undersubscribed in M'. Moreover, we have that the lecturer who offers p_j in I' (i.e., w) is either undersubscribed in M' or is full in M' with a student assigned to a project in Q. Further, u_i^z is either unassigned in M or prefers p_j to her assigned project in M, and the same is true for u_i^z in M'. Hence (u_i^z, p_j) is a blocking pair of M' in I', a contradiction.

Next, suppose (u_i^1, t_i) is a blocking pair of M in I, for some $i\,(1 \le i \le n_1)$. This implies that u_i^1 is unassigned in M and thus u_i^1 is unassigned in M'. Moreover, t_i is undersubscribed in M, and thus t_i is undersubscribed in M'. It follows that the lecturer who offers t_i in I' (i.e., x) is undersubscribed in M'. Hence (u_i^1, t_i) is a blocking pair of M' in I', a contradiction. Hence, no student in $U^1 \cup U^2$ can be involved in a blocking pair of M in I.

Next, suppose (s_i, q_i) forms a blocking pair for M in I, for some $i\,(1 \le i \le n_2)$. This implies that s_i is unassigned in M. Moreover, we have that each of q_i and w_i is undersubscribed in M. This implies that p_i is undersubscribed in M, and thus p_i is undersubscribed in M'. Hence, by the construction of M from M', it follows that $(s_i, q_i) \in M$, a contradiction.

The argument that M is coalition-free in I is similar to that given for M' in Proposition 2. Hence M is stable in I, and clearly $|M'| = |M|$. □

Propositions 2 and 3 imply that the reduction in [17, Section 3] holds with two lecturers involved. Given that it is NP-hard to approximate MAX-SPA-P within δ, for some constant $\delta > 1$, the following result is immediate.

Theorem 4. MAX-SPA-P-L2 *is NP-hard. Moreover, it is NP-hard to approximate* MAX-SPA-P-L2 *within δ, for some constant $\delta > 1$. The result holds even if each project has capacity* 1.

4. NP-hardness of (3, 3)-MAX-SPA-P

Recall that (3, 3)-MAX-SPA-P is the restriction of MAX-SPA-P in which each preference list is of length at most 3. In this section, we show that (3, 3)-MAX-SPA-P is NP-hard. The result holds even if each project and lecturer has capacity 1. To achieve this, we will show that (3, 3)-COM-SPA-P is NP-complete (this is the problem of deciding, given an instance of SPA-P in which all the preference lists are of length at most 3, whether a *complete stable matching*[3] exists). Clearly, the NP-completeness of (3, 3)-COM-SPA-P implies the NP-hardness of (3, 3)-MAX-SPA-P.

In order to prove this result, we use a reduction from a restricted version of SAT, which we define as follows. Let (2, 2)-E3-SAT denote the problem of deciding, given a Boolean formula B in CNF in which each clause contains exactly 3 literals and, for each variable v_i, each of literals v_i and $\overline{v_i}$ appears exactly twice in B, whether B is satisfiable. Berman et al. [3] showed that (2, 2)-E3-SAT is NP-complete.

Theorem 5. (3, 3)-COM-SPA-P *is NP-complete. The result holds even if each project and lecturer has capacity* 1.

Proof. Suppose we are given an assignment M in an arbitrary instance of (3, 3)-COM-SPA-P, clearly we can verify in polynomial-time if M is a complete stable matching. Hence (3, 3)-COM-SPA-P is in NP. Let B be an instance of (2, 2)-E3-SAT. Let $V = \{v_0, v_1, \ldots, v_{n-1}\}$ and $C = \{c_1, c_2, \ldots, c_m\}$ be the set of variables and clauses respectively in B. Then for each $v_i \in V$, each of literals v_i and $\overline{v_i}$ appears exactly twice in B. Also, for each $c_j \in C$, $|c_j| = 3$. Hence $m = \frac{4n}{3}$.

We construct an instance I of (3, 3)-COM-SPA-P as follows. Let $X \cup D \cup U$ be the set of students, where $X = \bigcup_{i=0}^{n-1} X_i$, $X_i = \{x_{4i+r} : 0 \le r \le 3\}\,(0 \le i \le n-1)$, $D = \bigcup_{j=1}^{m} D_j$, $D_j = \{d_j^1, d_j^2, d_j^3\}\,(1 \le j \le m)$ and $U = \{u_j : c_j \in C\}$. Let $Y \cup Z \cup P \cup Q \cup C'$ be the set of projects, where $Y = \bigcup_{i=0}^{n-1} Y_i$, $Y_i = \{y_{4i+r} : 0 \le r \le 3\}\,(0 \le i \le n-1)$, $Z = \{z_j : c_j \in C\}$,

[3] A stable matching in which every student is assigned.

Students' preferences		
x_{4i}:	$y_{4i}\ c(x_{4i})\ \ y_{4i+1}$	$(0 \le i \le n-1)$
x_{4i+1}:	$y_{4i+1}\ c(x_{4i+1})\ \ y_{4i+2}$	$(0 \le i \le n-1)$
x_{4i+2}:	$y_{4i+3}\ c(x_{4i+2})\ \ y_{4i+2}$	$(0 \le i \le n-1)$
x_{4i+3}:	$y_{4i}\ c(x_{4i+3})\ \ y_{4i+3}$	$(0 \le i \le n-1)$
d_j^s:	$z_j\ p_j^s$	$(1 \le j \le m \wedge 1 \le s \le 3)$
u_j:	$q_j^1\ q_j^2\ \ q_j^3$	$(1 \le j \le m)$
Lecturers' preferences		
l_j^s:	$p_j^s\ \ c_j^s\ \ q_j^s$	$(1 \le j \le m \wedge 1 \le s \le 3)$
y_r':	y_r	$(0 \le r \le 4n-1)$
z_j':	z_j	$(1 \le j \le m)$

Fig. 4. Preference lists for constructed instance of (3, 3)-COM-SPA-P.

$P=\bigcup_{j=1}^{m} P_j$, $P_j=\{p_j^1,p_j^2,p_j^3\}\,(1\le j\le m)$, $Q=\{q_j^s : c_j\in C \wedge 1\le s\le 3\}$ and $C'=\{c_j^s : c_j\in C\wedge 1\le s\le 3\}$. Let $L\cup Y'\cup Z'$ be the set of lecturers, where $L=\{l_j^s : c_j\in C\wedge 1\le s\le 3\}$ and offers $P\cup C'\cup Q$; $Y'=\{y_r' : 0\le r\le 4n-1\}$ and offers Y; and $Z'=\{z_j' : 1\le j\le m\}$ and offers Z. Finally, each project and lecturer has capacity 1. The preference lists in I are shown in Fig. 4.

In the preference list of a student $x_{4i+r}\in X\,(0\le i\le n-1)$, if $r\in\{0,1\}$ then the symbol $c(x_{4i+r})$ denotes the project $c_j^s\in C'$ such that the $(r+1)$th occurrence of literal v_i appears at position s of c_j. Similarly, if $r\in\{2,3\}$ then the symbol $c(x_{4i+r})$ denotes the project $c_j^s\in C'$ such that the $(r-1)$th occurrence of literal $\overline{v_i}$ appears at position s of c_j. We also use $x(c_j^s)$ to denote x_{4i+r} for $r\in\{0,1,2,3\}$. Clearly each preference list is of length at most 3.

For each $i\,(0\le i\le n-1)$, let $T_i=\{(x_{4i+r},y_{4i+r}) : 0\le r\le 3\}$ and $F_i=\{(x_{4i+r},y_{4i+r+1}) : 0\le r\le 3\}$, where addition is taken modulo 4. We claim that B is satisfiable if and only if I admits a complete stable matching.

Let f be a satisfying truth assignment of B. Define a complete stable matching M in I as follows. For each variable $v_i\in V$, if v_i is true under f, add the pairs in T_i to M; otherwise, add the pairs in F_i to M. Now let $c_j\in C$. As c_j contains a literal that is true under f, let $s\in\{1,2,3\}$ denote the position of c_j in which this literal occurs. Add the pairs (u_j,q_j^s), (d_j^s,z_j), (d_j^a,p_j^a) and (d_j^b,p_j^b) to M, where $\{a,b\}=\{1,2,3\}\setminus\{s\}$ and $a<b$. Clearly M is a complete matching in I.

No project in $Y\cup Z$ can be involved in a blocking pair of M, since each member of $Y\cup Z$ is full in M. Hence, no student in D can be involved in a blocking pair since any such student can only potentially prefer a project in Z. Similarly, no project in Q can be involved in a blocking pair of M since each lecturer $l_j^s\in L$ is full in M and since l_j^s either prefers her worst non-empty project to q_j^s, or q_j^s is her worst non-empty project. Hence no student in U can be involved in a blocking pair of M. Now suppose that $(x_{4i+r},c(x_{4i+r}))$ blocks M, where $0\le i\le n-1$ and $0\le r\le 3$. Let $c_j^s=c(x_{4i+r})$, where $1\le j\le m$ and $1\le s\le 3$. Then l_j^s is full in M and $(u_j,q_j^s)\in M$. If $r\in\{0,1\}$ then $(x_{4i+r},y_{4i+r+1})\in M$, so that v_i is false under f. But literal v_i occurs (unnegated) in c_j at position s, a contradiction, since literal v_i was supposed to be true under f by construction of M (since $(u_j,q_j^s)\in M$ if and only if c_j^s is true). Hence $r\in\{2,3\}$ and $(x_{4i+r},y_{4i+r})\in M$, so that v_i is true under f. But literal $\overline{v_i}$ occurs in c_j at position s, a contradiction, since $\overline{v_i}$ was supposed to be true under f by construction of M. Hence M admits no blocking pair.

We next verify that M is coalition-free. Clearly, no student in U can be in a coalition, since no two assigned students in U find the same project acceptable. Also, no student in D who is assigned in M to a project in Z can be in a coalition, since every such student is assigned to her first choice project. As a consequence, no student in D who is assigned in M to a project in P can be in a coalition, since each such student prefers only a project in Z. For each $i\,(0\le i\le n-1)$, no student in X_i can be in a coalition; for if $M\cap(X_i\times Y_i)=T_i$ then neither x_{4i} nor x_{4i+1} can be involved in a coalition, since each one of them is assigned in M to her first choice project. As a consequence, x_{4i+3} cannot be in a coalition, since the only student x_{4i+3} can potentially form a coalition with is x_{4_i}; and thus x_{4i+2} cannot be in a coalition, since the only student x_{4i+2} can potentially form a coalition with is x_{4i+3}. A similar argument can be made if $M\cap(X_i\times Y_i)=F_i$. Hence M is stable.

Conversely, suppose that M is a complete stable matching in I. Firstly, we claim that every project in C' is undersubscribed in M. To see this, observe that for each $j\,(1\le j\le m)$, u_j is assigned in M to some $q_j^s\,(1\le s\le 3)$. As a consequence, the three members of D_j can only be assigned to the three members of $\{z_j\}\cup(P_j\setminus\{p_j^s\})$, since each lecturer in L has capacity 1. Hence, for each $s\,(1\le s\le 3)$, c_j^s is not assigned to any student. Next, for each $i\,(0\le i\le n-1)$, we claim that $M\cap(X_i\times Y_i)$ is a perfect matching of $X_i\cup Y_i$. For suppose otherwise, then either (i) some student $x_{4i+r}\in X$ for $r\in\{0,1,2,3\}$ is unassigned in M, since every project in C' is undersubscribed in M, or (ii) some project $y_{4i+r}\in Y$ for

$r \in \{0, 1, 2, 3\}$ is undersubscribed in M. If (i) holds, we arrive at a contradiction, since M is a complete stable matching. If (ii) holds, we reach a contradiction following the argument from (i). We form a truth assignment f in B as follows. If $M \cap (X_i \times Y_i) = T_i$, set v_i to be true under f. Otherwise $M \cap (X_i \times Y_i) = F_i$, in which case we set v_i to be false under f.

Now let c_j be a clause in $C\,(1 \leq j \leq m)$. There exists some $s\,(1 \leq s \leq 3)$ such that $(u_j, q_j^s) \in M$. Let $x_{4i+r} = x(c_j^s)$ for some $i\,(0 \leq i \leq n-1)$ and $r\,(0 \leq r \leq 3)$. If $r \in \{0, 1\}$ then $(x_{4i+r}, y_{4i+r}) \in M$, since M is stable. Thus variable v_i is true under f and hence clause c_j is true under f, since literal v_i occurs unnegated in c_j. If $r \in \{2, 3\}$ then $(x_{4i+r}, y_{4i+r+1}) \in M$ (where addition is taken modulo 4), since M is stable. Thus variable v_i is false under f, and hence clause c_j is true under f, since literal $\overline{v_i}$ occurs in c_j. Hence f is a satisfying truth assignment of B. □

5. An IP model for MAX-SPA-P

Let I be an instance of SPA-P involving a set $\mathcal{S} = \{s_1, s_2, \ldots, s_{n_1}\}$ of students, a set $\mathcal{P} = \{p_1, p_2, \ldots, p_{n_2}\}$ of projects and a set $\mathcal{L} = \{l_1, l_2, \ldots, l_{n_3}\}$ of lecturers. We construct an IP model J of I as follows. Firstly, we create binary variables $x_{i,j} \in \{0, 1\}\,(1 \leq i \leq n_1, 1 \leq j \leq n_2)$ for each acceptable pair $(s_i, p_j) \in \mathcal{S} \times \mathcal{P}$ such that $x_{i,j}$ indicates whether s_i is assigned to p_j in a solution or not. Henceforth, we denote by S a solution in the IP model J, and we denote by M the matching derived from S. If $x_{i,j} = 1$ under S then intuitively s_i is assigned to p_j in M, otherwise s_i is not assigned to p_j in M.

5.1. Constraints

In this section, we give the set of constraints to ensure that the assignment obtained from a feasible solution in J is a matching, and that the matching admits no blocking pair and is coalition-free.

Matching constraints. The feasibility of a matching can be ensured with the following three set of constraints.

$$\sum_{p_j \in A_i} x_{i,j} \leq 1 \qquad (1 \leq i \leq n_1), \tag{1}$$

$$\sum_{i=1}^{n_1} x_{i,j} \leq c_j \qquad (1 \leq j \leq n_2), \tag{2}$$

$$\sum_{i=1}^{n_1} \sum_{p_j \in P_k} x_{i,j} \leq d_k \qquad (1 \leq k \leq n_3)\ . \tag{3}$$

Note that Inequality (1) ensures that each student $s_i \in \mathcal{S}$ is not assigned to more than one project, while Inequalities (2) and (3) ensure that the capacity of each project $p_j \in \mathcal{P}$ and each lecturer $l_k \in \mathcal{L}$ is not exceeded.

Given an acceptable pair (s_i, p_j), we define rank(s_i, p_j), the *rank* of p_j on s_i's preference list, to be $r+1$ where r is the number of projects that s_i prefers to p_j. Given a lecturer $l_k \in \mathcal{L}$ and a project $p_j \in P_k$, an analogous definition holds for rank(l_k, p_j), the *rank* of p_j on l_k's preference list. With respect to an acceptable pair (s_i, p_j), we define $S_{i,j} = \{p_{j'} \in A_i : \text{rank}(s_i, p_{j'}) \leq \text{rank}(s_i, p_j)\}$, the set of projects that s_i likes as much as p_j. For a project p_j offered by lecturer $l_k \in \mathcal{L}$, we also define $T_{k,j} = \{p_q \in P_k : \text{rank}(l_k, p_j) < \text{rank}(l_k, p_q)\}$, the set of projects that are worse than p_j on l_k's preference list.

In what follows, we fix an arbitrary acceptable pair (s_i, p_j) and we impose constraints to ensure that (s_i, p_j) is not a blocking pair for the matching M (i.e., (s_i, p_j) is not a type (a), type (b) or type (c) blocking pair for M). Firstly, let l_k be the lecturer who offers p_j.

Blocking pair constraints. We define $\theta_{i,j} = 1 - \sum_{p_{j'} \in S_{i,j}} x_{i,j'}$. Intuitively, $\theta_{i,j} = 1$ if and only if s_i is unassigned in M or prefers p_j to $M(s_i)$. Next we create a binary variable α_j in J such that if p_j is undersubscribed in M then $\alpha_j = 1$. We enforce this condition by imposing the following constraint.

$$c_j \alpha_j \geq c_j - \sum_{i'=1}^{n_1} x_{i',j}\ , \tag{4}$$

where $\sum_{i'=1}^{n_1} x_{i',j} = |M(p_j)|$. If p_j is undersubscribed in M then the RHS of Inequality (4) is at least 1, and this implies that $\alpha_j = 1$; otherwise, α_j is not constrained. Now let $\gamma_{i,j,k} = \sum_{p_{j'} \in T_{k,j}} x_{i,j'}$. Intuitively, if $\gamma_{i,j,k} = 1$ in S then s_i is assigned to a project $p_{j'}$ offered by l_k in M, where l_k prefers p_j to $p_{j'}$. The following constraint ensures that (s_i, p_j) does not form a type (a) blocking pair for M.

$$\theta_{i,j} + \alpha_j + \gamma_{i,j,k} \leq 2\ . \tag{5}$$

Note that if the sum of the binary variables in the LHS of Inequality (5) is less than or equal to 2, this implies that at least one of the variables, say $\gamma_{i,j,k}$, is 0. Thus the pair (s_i, p_j) is not a type (a) blocking pair for M.

Next we define $\beta_{i,k} = \sum_{p_{j'} \in P_k} x_{i,j'}$. Clearly, s_i is assigned to a project offered by l_k in M if and only if $\beta_{i,k} = 1$ in S. We define $D_{k,j} = \{p_{j'} \in P_k : \text{rank}(l_k, p_{j'}) \leq \text{rank}(l_k, p_j)\}$, the set of projects that l_k likes as much as p_j. Next, we create a binary

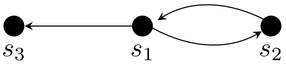

Fig. 5. The envy graph $G(M_2)$ corresponding to the SPA-P instance in Fig. 1.

variable $\eta_{j,k}$ in J such that $\eta_{j,k} = 1$ if l_k is undersubscribed or prefers p_j to his worst non-empty project in M. We enforce this by imposing the following constraint.

$$d_k \eta_{j,k} \geq d_k - \sum_{i'=1}^{n_1} \sum_{p_{j'} \in D_{k,j}} x_{i',j'} \ , \tag{6}$$

where $\sum_{i'=1}^{n_1} \sum_{p_{j'} \in D_{k,j}} x_{i',j'}$ is the occupancy of l_k involving projects that are at least as good as p_j on l_k's preference list. Intuitively if l_k is undersubscribed in M or if l_k prefers p_j to her worst non-empty project in M, then the RHS of Inequality (6) is at least 1. Finally, to avoid type (b) and type (c) blocking pairs, we impose the following constraint.

$$\theta_{i,j} + \alpha_j + (1 - \beta_{i,k}) + \eta_{j,k} \leq 3 \ . \tag{7}$$

Next, we give the constraints to ensure that the matching obtained from a feasible solution in J is coalition-free.

Coalition constraints. First, we introduce some additional notation. Given an instance I' of SPA-P and a matching M' in I', we define the *envy graph* $G(M') = (\mathcal{S}, A)$, where the vertex set $\mathcal{S}$ is the set of students in I', and the arc set

$$A = \{(s_i, s_{i'}) : s_i \text{ prefers } M'(s_{i'}) \text{ to } M'(s_i)\}.$$

It is clear that the matching $M_2 = \{(s_1, p_1), (s_2, p_2), (s_3, p_3)\}$ admits a coalition $\{s_1, s_2\}$ with respect to the instance given in Fig. 1. The resulting envy graph $G(M_2)$ is illustrated in Fig. 5.

Clearly, $G(M')$ contains a directed cycle if and only if M' admits a coalition. Moreover, $G(M')$ is acyclic if and only if it admits a topological ordering. Now to ensure that the matching M obtained from a feasible solution S under J is coalition-free, we will enforce J to encode the envy graph $G(M)$ and impose the condition that it must admit a topological ordering. In what follows, we build on our IP model J of I.

We create a binary variable $e_{i,i'}$ for each $(s_i, s_{i'}) \in \mathcal{S} \times \mathcal{S}$, $s_i \neq s_{i'}$, such that the $e_{i,i'}$ variables will correspond to the adjacency matrix of $G(M)$. For each i and i' $(1 \leq i \leq n_1,\ 1 \leq i' \leq n_1,\ i \neq i')$ and for each j and j' $(1 \leq j \leq n_2,\ 1 \leq j' \leq n_2)$ such that s_i prefers $p_{j'}$ to p_j, we impose the following constraint:

$$e_{i,i'} + 1 \geq x_{i,j} + x_{i',j'} \ . \tag{8}$$

If $(s_i, p_j) \in M$ and $(s_{i'}, p_{j'}) \in M$ and s_i prefers $p_{j'}$ to p_j, then $e_{i,i'} = 1$ and we say s_i *envies* $s_{i'}$; otherwise, $e_{i,i'}$ is not constrained. Next we enforce the condition that $G(M)$ must have a topological ordering. To hold the label of each vertex in a topological ordering, we create an integer-valued variable v_i corresponding to each student $s_i \in \mathcal{S}$ (and intuitively to each vertex in $G(M)$). We wish to enforce the constraint that if $e_{i,i'} = 1$ (i.e., $(s_i, s_{i'}) \in A$), then $v_i < v_{i'}$ (i.e., the label of vertex s_i is smaller than the label of vertex $s_{i'}$). This is achieved by imposing the following constraint for all i and i' $(1 \leq i \leq n_1,\ 1 \leq i' \leq n_1,\ i \neq i')$.

$$v_i + 1 \leq v_{i'} + n_1(1 - e_{i,i'}) \ . \tag{9}$$

Note that the LHS of Inequality (9) is strictly less than the RHS of Inequality (9) for all i, i $(i \neq i)$ if and only if $G(M)$ does not admit a directed cycle, and this implies that M is coalition-free.

5.2. Variables

We define a collective notation for each integer variable involved in J as follows.

$$\begin{aligned} X &= \{x_{i,j} : s_i \in \mathcal{S} \text{ and } p_j \in A_i\}, & \Lambda &= \{\alpha_j : 1 \leq j \leq n_2\}, \\ H &= \{\eta_{j,k} : 1 \leq j \leq n_2, 1 \leq k \leq n_3\}, & \Delta &= \{\delta_k : 1 \leq k \leq n_3\}, \\ E &= \{e_{i,i'} : 1 \leq i \leq n_1, 1 \leq i' \leq n_1\}, & V &= \{v_i : 1 \leq i \leq n_1\} \ . \end{aligned}$$

5.3. Objective function

The objective function given below is a summation of all the $x_{i,j}$ binary variables. It seeks to maximise the number of students assigned (i.e., the size of the matching).

$$\max \sum_{i=1}^{n_1} \sum_{p_j \in A_i} x_{i,j} \ . \tag{10}$$

Finally, we have constructed an IP model J of I comprising the set of integer-valued variables X, Λ, H, E and V, the set of Inequalities (1)–(9) and an objective function (10). Note that J can then be used to solve MAX-SPA-P optimally. Moreover, if J admits a feasible solution S, then the objective value denoted $obj(S)$ is equivalent to the number of students that are assigned in M, i.e., $obj(S) = |M|$.

5.4. Correctness of the IP model

Given an instance I of SPA-P formulated as an IP model J using the above transformation, we establish the correctness of J via the following lemmas.

Lemma 6. *A feasible solution S to J corresponds to a stable matching M in I, where $obj(S) = |M|$.*

Proof. Assume firstly that J has a feasible solution S. Let $M = \{(s_i, p_j) \in \mathcal{S} \times \mathcal{P} : x_{i,j} = 1\}$ be the assignment in I generated from S. Clearly $obj(S) = |M|$. We note that Inequality (1) ensures that each student is assigned in M to at most one project. Moreover, Inequalities (2) and (3) ensures that the capacity of each project and lecturer is not exceeded in M. Thus M is a matching. We will prove that Inequalities (4)–(7) guarantee that M admits no blocking pair.

Suppose for a contradiction that there exists some acceptable pair (s_i, p_j) that forms a blocking pair for M, where l_k is the lecturer who offers p_j. This implies that s_i is either unassigned in M or prefers p_j to $M(s_i)$. In either of these cases, $\sum_{p_{j'} \in S_{i,j}} x_{i,j'} = 0$, and thus $\theta_{i,j} = 1$. Moreover, as (s_i, p_j) is a blocking pair for M, p_j has to be undersubscribed in M, and thus $\sum_{i'=1}^{n_1} x_{i',j} < c_j$. This implies that the RHS of Inequality (4) is strictly greater than 0, and since S is a feasible solution to J, $\alpha_j = 1$.

Now suppose (s_i, p_j) is a type (a) blocking pair, and suppose $M(s_i) = p_{j''}$ for some $p_{j''} \in P_k$. We have that l_k prefers p_j to $p_{j''}$, thus $\gamma_{i,j,k} = \sum_{p_{j'} \in T_{k,j}} x_{i,j'} = 1$. Now, $\theta_{i,j} = \alpha_j = \gamma_{i,j,k} = 1$ implies that the LHS of Inequality (5) is strictly greater than 2. Thus S is not a feasible solution, a contradiction.

Next suppose (s_i, p_j) is a type (b) or type (c) blocking pair for M. This implies that $s_i \notin M(l_k)$ and thus $1 - \beta_{i,k} = 1 - \sum_{p_j' \in P_k} x_{i,j'} = 1$. Also, either l_k is undersubscribed in M or l_k prefers p_j to p_z, where p_z is l_k's worst non-empty project in M. This implies that the RHS of Inequality (6) is strictly greater than 0, and thus $\eta_{j,k} = 1$. Hence the LHS of Inequality (7) is strictly greater than 3. Thus S is not a feasible solution, a contradiction.

Finally, we show that Inequalities (8) and (9) ensure that M is coalition-free. Suppose for a contradiction that M admits a coalition $\langle s_{i_0}, \ldots, s_{i_{r-1}} \rangle$, for some $r \geq 2$. This implies that for each t $(0 \leq t \leq r-1)$, s_{i_t} prefers $M(s_{i_{t+1}})$ to $M(s_{i_t})$, where addition is taken modulo r, and hence $e_{i_t, i_{t+1}} = 1$, by Inequality (8). It follows from Inequality (9) that

$$v_{i_0} + 1 \leq v_{i_1} + 1 \leq \cdots \leq v_{i_{r-2}} + 1 \leq v_{i_{r-1}} + 1 \leq v_{i_r} = v_{i_0},$$

a contradiction. Hence M is coalition-free, and thus M is a stable matching. □

Lemma 7. *A stable matching M in I corresponds to a feasible solution S to J, where $|M| = obj(S)$.*

Proof. Let M be a stable matching in I. First we set all the binary variables involved in J to 0. For all $(s_i, p_j) \in M$, we set $x_{i,j} = 1$. Now, since M is a matching, it is clear that Inequalities (1)–(3) are satisfied. For any acceptable pair $(s_i, p_j) \in (\mathcal{S} \times \mathcal{P}) \setminus M$ such that s_i is unassigned in M or prefers p_j to $M(s_i)$, we set $\theta_{i,j} = 1$. For any project $p_j \in \mathcal{P}$ that is undersubscribed in M, we set $\alpha_j = 1$ and thus Inequality (4) is satisfied. For Inequality (5) not to be satisfied, its LHS must be strictly greater than 2. This would only happen if there exists $(s_i, p_j) \in (\mathcal{S} \times \mathcal{P}) \setminus M$, where l_k is the lecturer who offers p_j, such that $\theta_{i,j} = 1$, $\alpha_j = 1$ and $\gamma_{i,j,k} = 1$. This implies that s_i is assigned in M to a project $p_{j'}$ offered by l_k such that s_i prefers p_j to $p_{j'}$, p_j is undersubscribed in M, and l_k prefers p_j to $p_{j'}$. Thus (s_i, p_j) is a type (a) blocking pair for M, a contradiction to the stability of M. Hence Inequality (5) is satisfied.

Suppose l_k is a lecturer in $\mathcal{L}$ and p_j is any project on l_k's preference list. Let p_z be l_k's worst non-empty project in M. If l_k is undersubscribed in M or l_k prefers p_j to p_z, we set $\eta_{j,k} = 1$. Then Inequality (6) is satisfied. Now suppose Inequality (7) is not satisfied. This would only happen if there exists $(s_i, p_j) \in (\mathcal{S} \times \mathcal{P}) \setminus M$, where l_k is the lecturer who offers p_j, such that $\theta_{i,j} = 1$, $\alpha_j = 1$, $\beta_{i,k} = 0$ and $\eta_{j,k} = 1$. This implies that either s_i is unassigned in M or prefers p_j to $M(s_i)$, $s_i \notin M(l_k)$, p_j is undersubscribed in M and either l_k is undersubscribed in M or l_k prefers p_j to his worst non-empty project in M. Thus (s_i, p_j) is either a type (b) or type (c) blocking pair for M, a contradiction to the stability of M. Hence Inequality (7) is satisfied.

We denote by $G(M) = (\mathcal{S}, A)$ the envy graph of M. Suppose s_i and $s_{i'}$ are any two distinct students in $\mathcal{S}$ such that $(s_i, p_j) \in M$, $(s_{i'}, p_{j'}) \in M$ and s_i prefers $p_{j'}$ to p_j (i.e., $(s_i, s_{i'}) \in A$), we set $e_{i,i'} = 1$. Thus Inequality (8) is satisfied. Since M is a stable matching, M is coalition-free. This implies that $G(M)$ is acyclic and has a topological ordering

$$\sigma : \mathcal{S} \to \{1, 2, \ldots, n_1\}\ .$$

For each i $(1 \leq i \leq n_1)$, let $v_i = \sigma(s_i)$. Now suppose Inequality (9) is not satisfied. This implies that there exist vertices s_i and $s_{i'}$ in $G(M)$ such that $v_i + 1 > v_{i'} + n_1(1 - e_{i,i'})$. This is only possible if $e_{i,i'} = 1$ since $1 \leq v_i \leq n_1$ and $1 \leq v_{i'} \leq n_1$. Hence $v_i + 1 > v_{i'}$, a contradiction to the fact that σ is a topological ordering of $G(M)$ (since $(s_i, s_{i'}) \in A$ implies that $v_i < v_{i'}$). Hence S, comprising the above assignment of values to the variables in $X \cup \Lambda \cup H \cup \Delta \cup E \cup V$, is a feasible solution to J; and clearly $|M| = obj(S)$. □

Lemmas 6 and 7 immediately give rise to the following theorem regarding the correctness of J.

Theorem 8. *A feasible solution to J is optimal if and only if the corresponding stable matching in I is of maximum cardinality.*

Proof. Let S be an optimal solution to J. Then by Lemma 6, S corresponds to a stable matching M in I such that $obj(S) = |M|$. Suppose M is not of maximum cardinality. Then there exists a stable matching M' in I such that $|M'| > |M|$. By Lemma 7, M' corresponds to a feasible solution S' to J such that $obj(S') = |M'| > |M| = obj(S)$. This is a contradiction, since S is an optimal solution to J. Hence M is a maximum size stable matching in I. Similarly, if M is a maximum size stable matching in I then M corresponds to an optimal solution S to J. □

6. Empirical evaluation

In this section we present results from an empirical evaluation that investigates how the sizes of the stable matchings produced by the approximation algorithms compares to those of the optimal solution obtained from our IP model, on SPA-P instances that are both randomly generated and derived from real datasets.

6.1. Experimental setup

When generating SPA-P instances, there are clearly several parameters that can be varied, such as the number of students, projects and lecturers; the length of the students' preference lists; as well as the total capacities of the projects and lecturers. For each range of values for the first two parameters, we generated a set of random SPA-P instances. In each set, we recorded the average size of a stable matching obtained from running the approximation algorithms and the IP model. Further, we considered the average time taken for the IP model to find an optimal solution.

Very broadly, the approximation algorithms involve a sequence of applications and deletions. The students apply to projects that they find acceptable, and when a project and/or lecturer becomes full, certain (student, project) pairs are deleted. By design, the approximation algorithms were randomised with respect to the sequence in which students apply to projects, and the choice of students to reject when projects and/or lecturers become full. In the light of this, for each dataset, we also ran the approximation algorithms 100 times and record the size of the largest stable matching obtained over these runs. Our experiments therefore involved five algorithms: the optimal IP-based algorithm, the two approximation algorithms run once, and the two approximation algorithms run 100 times.

We performed our experiments on a machine with dual Intel Xeon CPU E5-2640 processors with 64GB of RAM, running Ubuntu 17.10. Each of the approximation algorithms was implemented in Java.[4] For our IP model, we carried out the implementation using the Gurobi optimisation solver in Java (see footnote 4). For correctness testing on these implementations, we designed a stability checker which verifies that the matching returned by the approximation algorithms and the IP model does not admit a blocking pair or a coalition.

6.2. Randomly-generated datasets

All the SPA-P instances that we randomly generated involved n_1 students (n_1 is henceforth referred to as the size of the instance), $0.5n_1$ projects, $0.2n_1$ lecturers and $1.1n_1$ total project capacity which was randomly distributed amongst the projects such that each project has capacity at least 1. The capacity for each lecturer l_k was chosen uniformly at random to lie between the highest capacity of the projects offered by l_k and the sum of the capacities of the projects that l_k offers. In the first experiment, we present results obtained from comparing the performance of the IP model, with and without the coalition constraints in place.

Experiment 0. We increased the number of students n_1 while maintaining a ratio of projects, lecturers, project capacities and lecturer capacities as described above. For various values of n_1 ($100 \leq n_1 \leq 1000$) in increments of 100, we created 100 randomly-generated instances. Each student's preference list contained a minimum of 2 and a maximum of 5 projects. With respect to each value of n_1, we obtained the average time taken for the IP solver to output a solution, both with and without the coalition constraints being enforced. The results, displayed in Table 1 show that when we removed the coalition constraints, the average time for the IP solver to output a solution is significantly faster than when we enforced the coalition constraints.

In the remaining experiments, we thus remove the constraints that enforce the absence of a coalition in the solution. We are able to do this for the purposes of these experiments because the largest size of a stable matching is equal to the largest size of a matching that potentially admits a coalition but admits no blocking pair,[5] and we were primarily concerned with measuring stable matching cardinalities. However the absence of the coalition constraints should be borne in mind when interpreting the IP solver runtime data in what follows.

In the next two experiments, we discuss results obtained from running the five algorithms on randomly-generated datasets.

[4] https://github.com/sofiatolaosebikan/spa-p-isco-2018

[5] This holds because the number of students assigned to each project and lecturer in the matching remains the same even after the students involved in such coalition permute their assigned projects.

Table 1
Results for Experiment 0. Average time (in seconds) for the IP solver to output a solution, both with and without the coalition constraints being enforced.

Size of instance	100	200	300	400	500	600	700	800	900	1000
Av. time without coalition	0.12	0.27	0.46	0.69	0.89	1.17	1.50	1.86	2.20	2.61
Av. time with coalition	0.71	2.43	4.84	9.15	13.15	19.34	28.36	38.18	48.48	63.50

Table 2
Properties of the real datasets and the size of stable matchings obtained from the five algorithms, with respect to Experiment 3, where A, B, C, D and E denote the solution obtained from the IP model, 100 runs of the $\frac{3}{2}$-approximation algorithm, single run of the $\frac{3}{2}$-approximation algorithm, 100 runs of the 2-approximation algorithm, and single run of the 2-approximation algorithm, respectively.

Year	n_1	n_2	n_3	l	Random					Most popular					Least popular				
					A	B	C	D	E	A	B	C	D	E	A	B	C	D	E
2014	55	149	38	6	55	55	55	54	53	55	55	55	54	50	55	55	55	54	52
2015	76	197	46	6	76	76	76	76	72	76	76	76	76	72	76	76	76	76	75
2016	92	214	44	6	84	82	83	77	75	85	85	83	79	76	82	80	77	76	74
2017	90	289	59	4	89	87	85	80	76	90	89	86	81	79	88	85	84	80	77

Experiment 1. As in the previous experiment, we maintained the ratio of the number of students to projects, lecturers and total project capacity, as well as the length of the students' preference lists. For various values of n_1 ($100 \leq n_1 \leq 2500$) in increments of 100, we created 1000 randomly-generated instances. With respect to each value of n_1, we obtained the average sizes of stable matchings constructed by the five algorithms run over the 1000 instances. The result displayed in Fig. 6(a) shows the ratio of the average size of the stable matching produced by the approximation algorithms with respect to the average size of the maximum cardinality stable matchings produced by the IP solver.

Fig. 6(a) shows that each of the approximation algorithms produces stable matchings with a much higher cardinality from multiple runs, compared to running them only once. Also, the average time taken for the IP solver to find a maximum cardinality matching increases as the size of the instance increases, with a running time of less than one second for instance size 100, increasing roughly linearly to 13 seconds for instance size 2500 (see Fig. 6(b)). Perhaps not surprising, each of the approximation algorithms terminates in less than one second for all the datasets.

Experiment 2. In this experiment, we varied the length of each student's preference list while maintaining a fixed number of students, projects, lecturers and total project capacity. For various values of x ($2 \leq x \leq 10$), we generated 1000 instances, each involving 1000 students, with each student's preference list containing exactly x projects. The result for all values of x is displayed in Fig. 7(a), which shows that as we increase the preference list length, the sizes of the stable matchings produced by each of the approximation algorithms approaches optimality. Fig. 7(a) also shows that with a preference list length greater than 5, the $\frac{3}{2}$-approximation algorithm produces an optimal solution, even on a single run. Moreover, the average time taken for the IP solver to find the size of a maximum size stable matching increases as the length of the students' preference lists increases, with a running time of two seconds when each student's preference list is of length 2, increasing roughly linearly to 17 seconds when each student's preference list is of length 10 (see Fig. 7(b)).

6.3. Real datasets

The real datasets in this chapter are based on actual student preference data and manufactured lecturer data from previous runs of student-project allocation processes at the School of Computing Science, University of Glasgow. Table 2 shows the properties of the real datasets, where n_1, n_2 and n_3 denote the number of students, projects and lecturers respectively; and l denotes the length of each student's preference list. For all these datasets, each project has a capacity of 1 and the capacity of each lecturer was provided as part of the datasets. In the next experiment, we discuss how the lecturer preferences over their proposed projects were generated (which is the only information we manufactured). We also discuss the results obtained from running the five algorithms on the corresponding SPA-P instances.

Experiment 3. We derived the lecturer preference data from the real datasets as follows. For each lecturer l_k, and for each project p_j offered by l_k, we obtained the number a_j of students that find p_j acceptable. Next, we generated a strictly-ordered preference list for l_k by arranging l_k's proposed projects in (i) a random manner, (ii) ascending order of a_j, and (iii) descending order of a_j, where (ii) and (iii) are taken over all projects that l_k offers. Table 2 shows the size of stable matchings obtained from the five algorithms, and the results are essentially consistent with the findings in the previous experiments: i.e., the $\frac{3}{2}$-approximation algorithm produces stable matchings whose sizes are close to optimal.

6.4. Discussions

The results presented in this section suggest that even as we increase the number of students, projects, lecturers, and the length of the students' preference lists, each of the approximation algorithms finds stable matchings that are close to

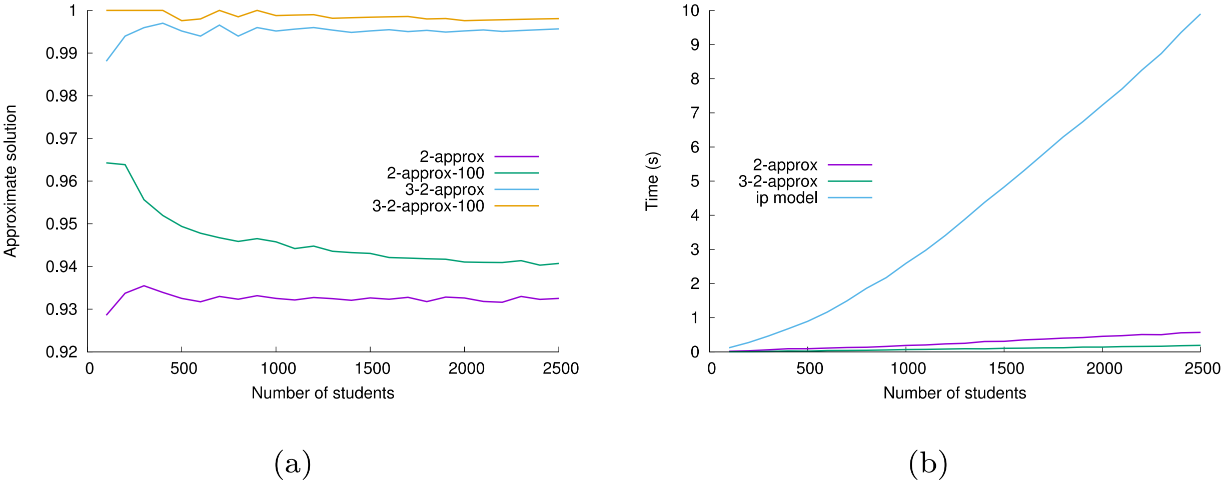

Fig. 6. Result for Experiment 1.

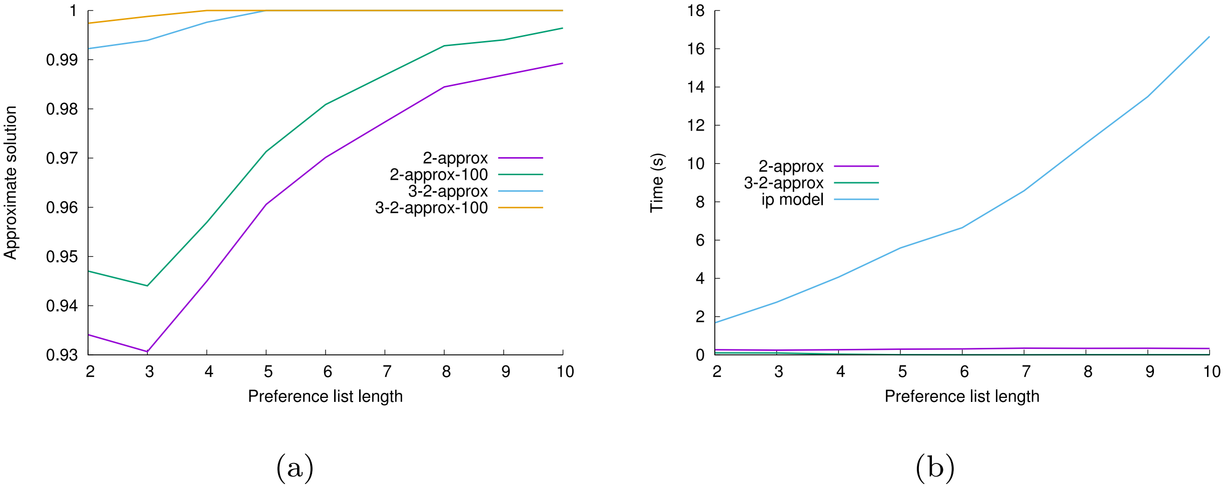

Fig. 7. Result for Experiment 2.

having maximum cardinality, outperforming their approximation factor. Perhaps most interesting is the $\frac{3}{2}$-approximation algorithm, which finds stable matchings that are very close in size to optimal, even on a single run. These results also hold analogously for the instances derived from real datasets.

We remark that when we removed the coalition constraints, we were able to run the IP model on an instance size of 10 000, with the solver returning the size of a maximum matching in an average time of 100 seconds, over 100 randomly-generated instances. This shows that the IP model (without enforcing the coalition constraints), can be run on large instances that could potentially appear in practical applications of the SPA-P model, to find maximum size stable matchings that potentially admits coalition of students. These coalitions should then be eliminated in polynomial time by repeatedly constructing an *envy graph*, similar to the one described in [15, p. 290], finding a directed cycle and letting the students in the cycle swap projects.

7. Conclusions and open problems

In this paper, we have presented algorithmic and experimental results for finding maximum size stable matchings in instances of SPA-P. From an algorithmic perspective, we have shown that MAX-SPA-P becomes polynomial-time solvable if there is only one lecturer, whilst the problem remains NP-hard to approximate even if there are two lecturers involved. We also proved that it is NP-hard to find a maximum size stable matching if each preference list is of length at most 3. It would be interesting to consider other polynomial-time solvable special cases, for example, what if each student's preference list is of length at most 2 and each lecturer's preference list is of unbounded length?

Moving away from hardness results, a particularly interesting direction would be to explore parameterisations that lead to FPTalgorithms for MAX-SPA-P. Whilst our NP-hardness result shows that parameterising on the number of lecturers or the maximum length of a preference list is not a good choice, other suitable parameterisations that could be explored include the maximum capacity of a project or lecturer, which we might expect to be small in practice. On the other hand, a different direction is to establish W[1]-hardness results under various parameterisations.

To enable MAX-SPA-P to be solved optimally in practice, we went on to describe an IP model for the problem. From our experimental results, we were able to deduce that the $\frac{3}{2}$-approximation algorithm of Iwama et al. [11] constructs stable matchings whose size is very close to optimal. Nevertheless, the question remains as to whether there exists an approximation algorithm for MAX-SPA-P that has a performance guarantee better than $\frac{3}{2}$?

Acknowledgments

The authors would like to thank Elias Koutsoupias for suggesting to consider the restriction of SPA-P in which the number of lecturers is a constant. We would also like to thank the anonymous reviewers for valuable comments that helped to improve the presentation of this paper.

References

[1] D.J. Abraham, R.W. Irving, D.F. Manlove, Two algorithms for the Student-Project allocation problem, J. Discrete Algorithms 5 (1) (2007) 79–91.
[2] A.A. Anwar, A.S. Bahaj, Student project allocation using integer programming, IEEE Trans. Educ. 46 (3) (2003) 359–367.
[3] P. Berman, M. Karpinski, Alexander D. Scott, Approximation Hardness of Short Symmetric Instances of MAX-3SAT, Electronic Colloquium on Computational Complexity Report, Number 49, 2003.
[4] R. Calvo-Serrano, G. Guillén-Gosálbez, S. Kohn, A. Masters, Mathematical programming approach for optimally allocating students' projects to academics in large cohorts, Educ. Chem. Eng. 20 (2017) 11–21.
[5] M. Chiarandini, R. Fagerberg, S. Gualandi, Handling preferences in student-project allocation, Ann. Oper. Res. 275 (1) (2019) 39–78.
[6] CPLEX Optimization Studio, 2020, http://www-03.ibm.com/software/products/en/ibmilogcpleoptistud/, (Accessed 25 2020).
[7] D. Gale, L.S. Shapley, College admissions and the stability of marriage, Amer. Math. Monthly 69 (1962) 9–15.
[8] GNU Linear Proramming Kit, 2020, https://www.gnu.org/software/glpk, (Accessed 25 2020).
[9] Gurobi Optimization website, 2020, http://www.gurobi.com, (Accessed 25 2020).
[10] P.R. Harper, V. de Senna, I.T. Vieira, A.K. Shahani, A genetic algorithm for the project assignment problem, Comput. Oper. Res. 32 (2005) 1255–1265.
[11] K. Iwama, S. Miyazaki, H. Yanagisawa, Improved approximation bounds for the student-project allocation problem with preferences over projects, J. Discrete Algorithms 13 (2012) 59–66.
[12] D. Kazakov, Co-ordination of student-project allocation, Manuscript, University of York, Department of Computer Science, 2001, Available from http://www-users.cs.york.ac.uk/kazakov/papers/proj.pdf (Accessed 25 2020).
[13] Z. Király, Better and simpler approximation algorithms for the stable marriage problem, Algorithmica 60 (2011) 3–20.
[14] A. Kwanashie, R.W. Irving, D.F. Manlove, C.T.S. Sng, Profile-based optimal matchings in the Student–Project Allocation problem, in: Proceedings of IWOCA 2014: The 25th International Workshop on Combinatorial Algorithms, in: Lecture Notes in Computer Science, vol. 8986, Springer, 2015, pp. 213–225.
[15] D.F. Manlove, Algorithmics of Matching under Preferences, World Scientific, 2013.
[16] D.F. Manlove, D. Milne, S. Olaosebikan, An integer programming approach to the student-project allocation problem with preferences over projects, in: Proceedings of ISCO '18: The 5th International Symposium on Combinatorial Optimization, in: Lecture Notes in Computer Science, vol. 10856, Springer, 2018, pp. 313–325.
[17] D.F. Manlove, G. O'Malley, Student project allocation with preferences over projects, J. Discrete Algorithms 6 (2008) 553–560.
[18] L.G. Proll, A simple method of assigning projects to students, Oper. Res. Q. 23 (2) (1972) 195–201.
[19] A.E. Roth, The evolution of the labor market for medical interns and residents: a case study in game theory, J. Polit. Econ. 92 (6) (1984) 991–1016.
[20] C.Y. Teo, D.J. Ho, A systematic approach to the implementation of final year project in an electrical engineering undergraduate course, IEEE Trans. Educ. 41 (1) (1998) 25–30.