An Integer Programming Formulation for a Matching Problem

David Manlove, Duncan Milne and Sofiat Olaosebikan

School of Computing Science, University of Glasgow

BCTCS 2018, Royal Holloway, University of London

March 28, 2018

Outline

1. Introduction
   - Matching Problems
   - Student-Project Allocation problem (SPA)
   - SPA with preferences over Projects (SPA-P)
   - The problem: MAX-SPA-P
2. An Integer Programming (IP) model for MAX-SPA-P
3. Experimental results
4. Discussions and Future work

Matching Problems

This class of problem generally involves

- assigning a set of agents to another set of agents

Matching Problems

This class of problem generally involves

- assigning a set of agents to another set of agents
- based on the preferences of the agents

Matching Problems

This class of problem generally involves

- assigning a set of agents to another set of agents
- based on the preferences of the agents
- and some problem-specific constraints

Matching Problems

This class of problem generally involves

- assigning a set of agents to another set of agents
- based on the preferences of the agents
- and some problem-specific constraints
    - for example, the capacity of the agents

Matching Problems

This class of problem generally involves

- assigning a set of agents to another set of agents
- based on the preferences of the agents
- and some problem-specific constraints
  - for example, the capacity of the agents

Example applications include

- allocation of junior doctors to hospitals

Matching Problems

This class of problem generally involves

- assigning a set of agents to another set of agents
- based on the preferences of the agents
- and some problem-specific constraints
  - for example, the capacity of the agents

Example applications include

- allocation of junior doctors to hospitals
- assigning conference papers to reviewers

Matching Problems

This class of problem generally involves

- assigning a set of agents to another set of agents
- based on the preferences of the agents
- and some problem-specific constraints
  - for example, the capacity of the agents

Example applications include

- allocation of junior doctors to hospitals
- assigning conference papers to reviewers
- assigning students to projects

Student-Project Allocation Problem (SPA)

SPA involves

- the assignment of students to projects offered by lecturers

Student-Project Allocation Problem (SPA)

SPA involves

- the assignment of students to projects offered by lecturers
- based on the capacities of projects and lecturers

Student-Project Allocation Problem (SPA)

SPA involves

- the assignment of students to projects offered by lecturers
- based on the capacities of projects and lecturers
- students' preferences over projects

Student-Project Allocation Problem (SPA)

SPA involves

- the assignment of students to projects offered by lecturers
- based on the capacities of projects and lecturers
- students' preferences over projects
- lecturers' preferences over
  - students (SPA-S), or

Student-Project Allocation Problem (SPA)

SPA involves

- the assignment of students to projects offered by lecturers
- based on the capacities of projects and lecturers
- students' preferences over projects
- lecturers' preferences over
  - students (SPA-S), or
  - projects (SPA-P), or

Student-Project Allocation Problem (SPA)

SPA involves

- the assignment of students to projects offered by lecturers
- based on the capacities of projects and lecturers
- students' preferences over projects
- lecturers' preferences over
  - students (SPA-S), or
  - projects (SPA-P), or
  - student-project pairs (SPA-(S,P))

Student-Project Allocation Problem (SPA)

SPA involves

- the assignment of students to projects offered by lecturers
- based on the capacities of projects and lecturers
- students' preferences over projects
- lecturers' preferences over
  - students (SPA-S), or
  - projects (SPA-P), or
  - student-project pairs (SPA-(S,P))

SPA with preferences over Projects (SPA-P)

Students' preferences	Lecturers' preferences
s_1: p_3 p_2 p_1	l_1: p_1 p_2
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: $d_1 = 2$, $d_2 = 1$.

SPA with preferences over Projects (SPA-P)

Students' preferences		Lecturers' preferences
s_1:	$p_3 \quad p_2 \quad p_1$	l_1: $p_1 \quad p_2$
s_2:	$p_1 \quad p_2$	l_2: p_3
s_3:	p_3	

Project capacities: $c_1 = c_2 = c_3 = 1$.
Lecturer capacities: $d_1 = 2,\ d_2 = 1$.

What we seek...

- a *matching* of students to projects based on these preferences

SPA with preferences over Projects (SPA-P)

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
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: $d_1 = 2$, $d_2 = 1$.

What we seek...

- a *matching* of students to projects based on these preferences
  - each student is not assigned more than one project

SPA with preferences over Projects (SPA-P)

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
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: $d_1 = 2,\ d_2 = 1$.

What we seek...

- a *matching* of students to projects based on these preferences
  - each student is not assigned more than one project
  - capacities of projects and lecturers are not exceeded

A matching..

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
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: $d_1 = 2,\ d_2 = 1$.

A matching..

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
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: $d_1 = 2,\ d_2 = 1$.

however,

- s_2 would prefer to be assigned p_1

A matching..

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
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: $d_1 = 2,\ d_2 = 1$.

however,

- s_2 would prefer to be assigned p_1
- this means l_1 also gets her most preferred project

A matching..

Students' preferences	Lecturers' preferences
s_1: p_3 p_2 p_1	l_1: p_1 p_2
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: $d_1 = 2,\ d_2 = 1$.

however,

- s_2 would prefer to be assigned p_1
- this means l_1 also gets her most preferred project
- we call (s_2, p_1) a blocking pair

Definition: Blocking Pair

Definition: Blocking Pair

Given an instance I of SPA-P, and a matching M in I. The pair (s_i, p_j) forms a *blocking pair* relative to M, where l_k is the lecturer who offers p_j, if:

Definition: Blocking Pair

Given an instance I of SPA-P, and a matching M in I. The pair (s_i, p_j) forms a *blocking pair* relative to M, where l_k is the lecturer who offers p_j, if:

1. either s_i is unassigned in M or s_i prefers p_j to $M(s_i)$, and

Definition: Blocking Pair

Given an instance I of SPA-P, and a matching M in I. The pair (s_i, p_j) forms a *blocking pair* relative to M, where l_k is the lecturer who offers p_j, if:

1. either s_i is unassigned in M or s_i prefers p_j to $M(s_i)$, and
2. p_j is undersubscribed in M, and either

Definition: Blocking Pair

Given an instance I of SPA-P, and a matching M in I. The pair (s_i, p_j) forms a *blocking pair* relative to M, where l_k is the lecturer who offers p_j, if:

1. either s_i is unassigned in M or s_i prefers p_j to $M(s_i)$, and
2. p_j is undersubscribed in M, and either
   (i) $s_i \in M(l_k)$ and l_k prefers p_j to $M(s_i)$, or

Definition: Blocking Pair

Given an instance I of SPA-P, and a matching M in I. The pair (s_i, p_j) forms a *blocking pair* relative to M, where l_k is the lecturer who offers p_j, if:

1. either s_i is unassigned in M or s_i prefers p_j to $M(s_i)$, and
2. p_j is undersubscribed in M, and either
   - (i) $s_i \in M(l_k)$ and l_k prefers p_j to $M(s_i)$, or
   - (ii) $s_i \notin M(l_k)$ and l_k is undersubcribed, or

Definition: Blocking Pair

Given an instance I of SPA-P, and a matching M in I. The pair (s_i, p_j) forms a *blocking pair* relative to M, where l_k is the lecturer who offers p_j, if:

1. either s_i is unassigned in M or s_i prefers p_j to $M(s_i)$, and
2. p_j is undersubscribed in M, and either
   - (i) $s_i \in M(l_k)$ and l_k prefers p_j to $M(s_i)$, or
   - (ii) $s_i \notin M(l_k)$ and l_k is undersubcribed, or
   - (iii) $s_i \notin M(l_k)$ and l_k prefers p_j to her worst non-empty project in $M(l_k)$.

Another matching..

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
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: $d_1 = 2,\ d_2 = 1$.

Another matching..

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
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: $d_1 = 2,\ d_2 = 1$.

- s_1 and s_2 would rather swap their assigned projects, in order to be better off

Another matching..

`Students' preferences`

s_1: p_3 p_2 p_1

s_2: p_1 p_2

s_3: p_3

`Lecturers' preferences`

l_1: p_1 p_2

l_2: p_3

Project capacities: $c_1 = c_2 = c_3 = 1$.

Lecturer capacities: $d_1 = 2$, $d_2 = 1$.

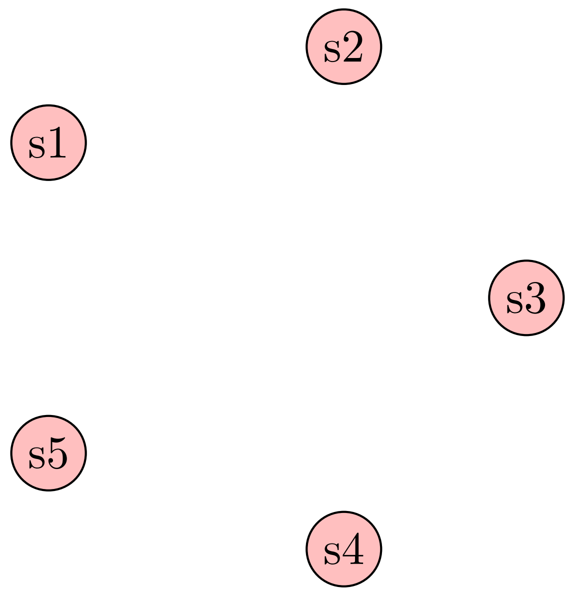
- s_1 and s_2 would rather swap their assigned projects, in order to be better off
- we call $\{s_1, s_2\}$ a coalition

Definition: Coalition

Given a matching M, a *coalition* is a set of students $\{s_{i_0}, \ldots, s_{i_{r-1}}\}$, 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.

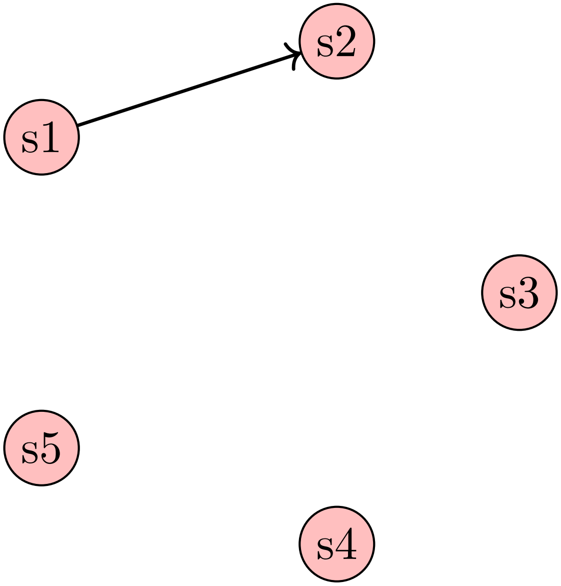
Definition: Coalition

Given a matching M, a *coalition* is a set of students $\{s_{i_0}, \ldots, s_{i_{r-1}}\}$, 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.



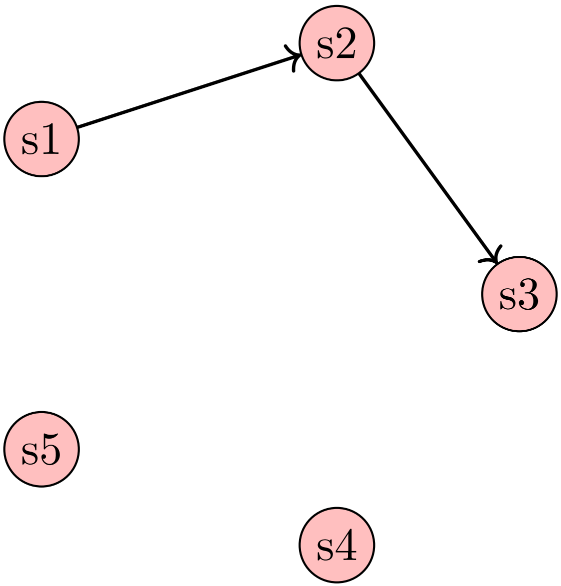
Definition: Coalition

Given a matching M, a *coalition* is a set of students $\{s_{i_0}, \ldots, s_{i_{r-1}}\}$, 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.



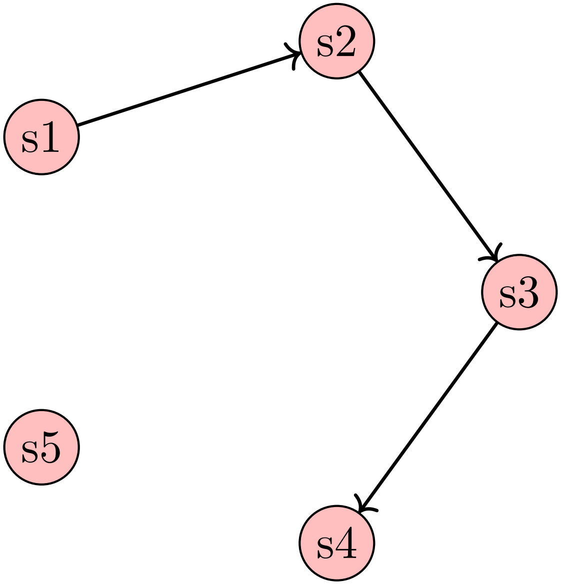
Definition: Coalition

Given a matching M, a *coalition* is a set of students $\{s_{i_0}, \ldots, s_{i_{r-1}}\}$, 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.



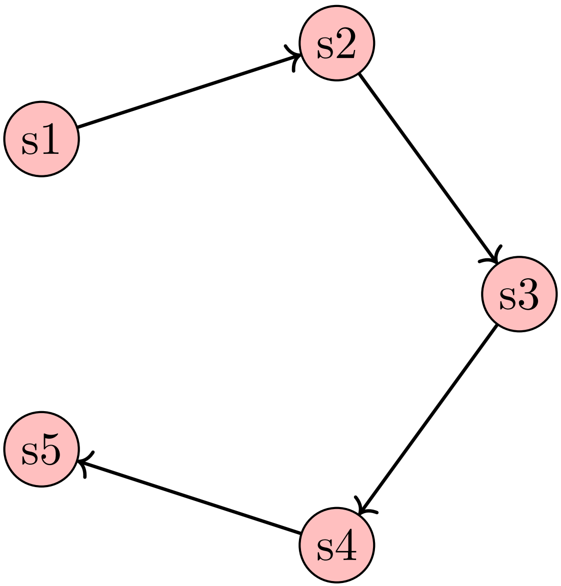
Definition: Coalition

Given a matching M, a *coalition* is a set of students $\{s_{i_0}, \ldots, s_{i_{r-1}}\}$, 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.



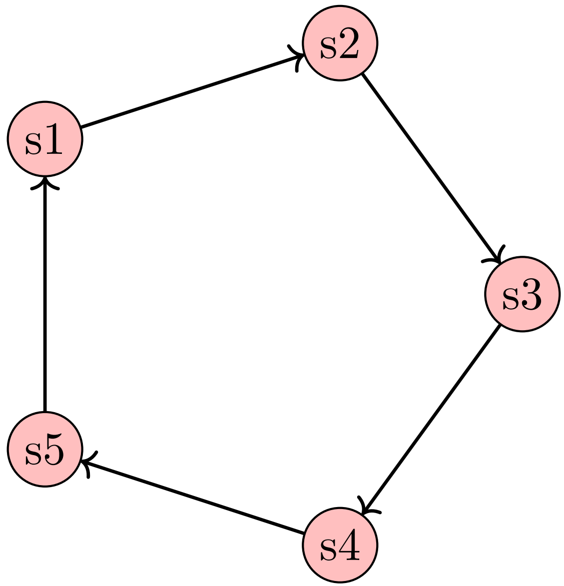
Definition: Coalition

Given a matching M, a *coalition* is a set of students $\{s_{i_0}, \ldots, s_{i_{r-1}}\}$, 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.



Definition: Coalition

Given a matching M, a *coalition* is a set of students $\{s_{i_0}, \ldots, s_{i_{r-1}}\}$, 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.



The type of matching we seek..

The type of matching we seek..

Stable matchings

- one with no blocking pair and no coalition

Image adapted from `https://bit.ly/2uBuuA0` (last accessed 28 March 2018).

Stable matchings..

A stable matching

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
s_2:	p_1	p_2		l_2:	p_3	
s_3:	p_3					

- 2 students are matched

Stable matchings..

A stable matching

Students' preferences	Lecturers' preferences
s_1: p_3 p_2 p_1	l_1: p_1 p_2
s_2: p_1 p_2	l_2: p_3
s_3: p_3	

- 2 students are matched

Another stable matching

Students' preferences	Lecturers' preferences
s_1: p_3 p_2 p_1	l_1: p_1 p_2
s_2: p_1 p_2	l_2: p_3
s_3: p_3	

- 3 students are matched

Maximum cardinality stable matching

Another problem..

- finding a maximum cardinality stable matching (MAX-SPA-P)

Maximum cardinality stable matching

Another problem..

- finding a maximum cardinality stable matching (MAX-SPA-P)
- MAX-SPA-P is NP-hard

Maximum cardinality stable matching

Another problem..

- finding a maximum cardinality stable matching (MAX-SPA-P)
- MAX-SPA-P is NP-hard

Existing results for MAX-SPA-P

Maximum cardinality stable matching

Another problem..

- finding a maximum cardinality stable matching (MAX-SPA-P)
- MAX-SPA-P is NP-hard

Existing results for MAX-SPA-P

Suppose the size of a maximum stable matching M is 12,

- 2-approximation algorithm[a], i.e., solution at least $\frac{1}{2}M = 6$

Maximum cardinality stable matching

Another problem..

- finding a maximum cardinality stable matching (MAX-SPA-P)
- MAX-SPA-P is NP-hard

Existing results for MAX-SPA-P

Suppose the size of a maximum stable matching M is 12,

- 2-approximation algorithm[a], i.e., solution at least $\frac{1}{2}M = 6$
- $\frac{3}{2}$-approximation algorithm[b], i.e., solution at least $\frac{2}{3}M = 8$
  - not approximable within $\frac{21}{19} - \epsilon$, for any $\epsilon > 0$, unless P = NP

[a] D.F. Manlove and G. O'Malley. Student project allocation with preferences over projects. Journal of Discrete Algorithms, 6:553–560, 2008

[b] K. Iwama, S. Miyazaki, and H. Yanagisawa. Improved approximation bounds for the student-project allocation problem with preferences over projects. Journal of Discrete Algorithms, 13:59–66, 2012.

An Integer Programming (IP) model for MAX-SPA-P

An Integer Programming (IP) model for MAX-SPA-P

A general construction of our IP model

An Integer Programming (IP) model for MAX-SPA-P

A general construction of our IP model

- create binary-valued variables to represent the assignment of students to projects;

An Integer Programming (IP) model for MAX-SPA-P

A general construction of our IP model

- create binary-valued variables to represent the assignment of students to projects;
- enforce the following classes of constraints:

An Integer Programming (IP) model for MAX-SPA-P

A general construction of our IP model

- create binary-valued variables to represent the assignment of students to projects;
- enforce the following classes of constraints:
  1. find a matching;

An Integer Programming (IP) model for MAX-SPA-P

A general construction of our IP model

- create binary-valued variables to represent the assignment of students to projects;
- enforce the following classes of constraints:
  1. find a matching;
  2. ensure matching does not admit a blocking pair;

An Integer Programming (IP) model for MAX-SPA-P

A general construction of our IP model

- create binary-valued variables to represent the assignment of students to projects;
- enforce the following classes of constraints:
  1. find a matching;
  2. ensure matching does not admit a blocking pair;
  3. ensure matching does not admit a coalition;

An Integer Programming (IP) model for MAX-SPA-P

A general construction of our IP model

- create binary-valued variables to represent the assignment of students to projects;
- enforce the following classes of constraints:
  1. find a matching;
  2. ensure matching does not admit a blocking pair;
  3. ensure matching does not admit a coalition;
- describe an objective function to maximise the size of the matching.

Encoding the binary-valued variables

Students' preferences		Lecturers' preferences	
s_1:	p_3 p_2 p_1	l_1:	p_1 p_2
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: $d_1 = 2,\ d_2 = 1$.

We encode each (s_i, p_j) as a variable $x_{i,j} \in \{0, 1\}$

Encoding the binary-valued variables

Students' preferences		Lecturers' preferences
s_1: p_3 p_2 p_1		l_1: p_1 p_2
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: $d_1 = 2$, $d_2 = 1$.

We encode each (s_i, p_j) as a variable $x_{i,j} \in \{0, 1\}$

$x_{1,3}$ $\quad x_{1,2}$ $\quad x_{1,1}$

Encoding the binary-valued variables

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
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: $d_1 = 2$, $d_2 = 1$.

We encode each (s_i, p_j) as a variable $x_{i,j} \in \{0, 1\}$

$x_{1,3}$ $\quad x_{1,2}$ $\quad x_{1,1}$

$\Downarrow$

$= 1$, then s_1 is assigned to p_3

Encoding the binary-valued variables

Students' preferences

s_1: p_3 p_2 p_1

s_2: p_1 p_2

s_3: p_3

Lecturers' preferences

l_1: p_1 p_2

l_2: p_3

Project capacities: $c_1 = c_2 = c_3 = 1$.

Lecturer capacities: $d_1 = 2,\ d_2 = 1$.

We encode each (s_i, p_j) as a variable $x_{i,j} \in \{0, 1\}$

$x_{1,3}$ $x_{1,2}$ $x_{1,1}$

$\Downarrow$

$= 1$, then s_1 is assigned to p_3

$= 0$, then s_1 is not assigned to p_3

Encoding the binary-valued variables

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
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: $d_1 = 2,\ d_2 = 1$.

We encode each (s_i, p_j) as a variable $x_{i,j} \in \{0, 1\}$

$x_{1,3}$ $\quad x_{1,2}$ $\quad x_{1,1}$

$\Downarrow$

$= 1$, then s_1 is assigned to p_3
$= 0$, then s_1 is not assigned to p_3

$x_{2,1}$ $\quad x_{2,2}$

Encoding the binary-valued variables

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
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: $d_1 = 2,\ d_2 = 1$.

We encode each (s_i, p_j) as a variable $x_{i,j} \in \{0, 1\}$

$x_{1,3}$ $x_{1,2}$ $x_{1,1}$

$\Downarrow$

$= 1$, then s_1 is assigned to p_3
$= 0$, then s_1 is not assigned to p_3

$x_{2,1}$ $x_{2,2}$

$x_{3,3}$

Matching Constraints

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
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: $d_1 = 2$, $d_2 = 1$.

Matching Constraints

Students' preferences	Lecturers' preferences
s_1: p_3 p_2 p_1	l_1: p_1 p_2
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: $d_1 = 2$, $d_2 = 1$.

- each student is not assigned more than one project

Matching Constraints

Students' preferences	Lecturers' preferences
s_1: p_3 p_2 p_1	l_1: p_1 p_2
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: $d_1 = 2,\ d_2 = 1$.

- each student is not assigned more than one project

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

Matching Constraints

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
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: $d_1 = 2,\ d_2 = 1$.

- each student is not assigned more than one project

$$\boxed{\sum_{p_j \in A_i} x_{i,j} \leq 1 \quad (1 \leq i \leq n_1), \qquad \Longrightarrow \quad x_{1,3} + x_{1,2} + x_{1,1} \leq 1}$$

Matching Constraints

Students' preferences	Lecturers' preferences
s_1: p_3 p_2 p_1	l_1: p_1 p_2
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: $d_1 = 2,\ d_2 = 1$.

- each student is not assigned more than one project

$$\boxed{\sum_{p_j \in A_i} x_{i,j} \leq 1 \quad (1 \leq i \leq n_1), \qquad \Longrightarrow \ x_{1,3} + x_{1,2} + x_{1,1} \leq 1}$$

- capacities of projects are not exceeded

Matching Constraints

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
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: $d_1 = 2,\ d_2 = 1$.

- each student is not assigned more than one project

$$\sum_{p_j \in A_i} x_{i,j} \leq 1 \quad (1 \leq i \leq n_1), \qquad \Longrightarrow \quad x_{1,3} + x_{1,2} + x_{1,1} \leq 1$$

- capacities of projects are not exceeded

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

Matching Constraints

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
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: $d_1 = 2,\ d_2 = 1$.

- each student is not assigned more than one project

$$\sum_{p_j \in A_i} x_{i,j} \leq 1 \quad (1 \leq i \leq n_1), \qquad \Longrightarrow \quad x_{1,3} + x_{1,2} + x_{1,1} \leq 1$$

- capacities of projects are not exceeded

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

Matching Constraints..

Students' preferences	Lecturers' preferences
s_1: p_3 p_2 p_1	l_1: p_1 p_2
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: $d_1 = 2,\ d_2 = 1$.

- capacities of lecturers are not exceeded

Matching Constraints..

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
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: $d_1 = 2$, $d_2 = 1$.

- capacities of lecturers are not exceeded

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

Matching Constraints..

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
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: $d_1 = 2$, $d_2 = 1$.

- capacities of lecturers are not exceeded

$$\sum_{i=1}^{n_1} \sum_{p_j \in P_k} x_{i,j} \leq d_k \qquad (1 \leq k \leq n_3),$$
$$\Longrightarrow \; x_{1,2} + x_{1,1} + x_{2,1} + x_{2,2} \leq 2$$

Blocking pair constraints

Blocking pair constraints

Students' preferences		Lecturers' preferences	
s_1:	p_3 p_2 p_1	l_1:	p_1 p_2
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: $d_1 = 2$, $d_2 = 1$.

Blocking pair constraints

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
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: $d_1 = 2$, $d_2 = 1$.

For each (s_i, p_j), where l_k is the lecturer who offers p_j, we

- define $\theta_{i,j} = 1 - \sum_{p_{j'} \in S_{i,j}} x_{i,j'}$

Blocking pair constraints

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
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: $d_1 = 2,\ d_2 = 1.$

For each (s_i, p_j), where l_k is the lecturer who offers p_j, we

- define $\theta_{i,j} = 1 - \sum_{p_{j'} \in S_{i,j}} x_{i,j'} \implies \theta_{2,1} = 1 - x_{2,1} = 1.$

Blocking pair constraints

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
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: $d_1 = 2,\ d_2 = 1$.

For each (s_i, p_j), where l_k is the lecturer who offers p_j, we

- define $\theta_{i,j} = 1 - \sum_{p_{j'} \in S_{i,j}} x_{i,j'} \implies \theta_{2,1} = 1 - x_{2,1} = 1$.
- create $\alpha_j \in \{0, 1\}$, enforce $c_j \alpha_j \geq c_j - \sum_{i'=1}^{n_1} x_{i',j}$

Blocking pair constraints

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
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: $d_1 = 2,\ d_2 = 1$.

For each (s_i, p_j), where l_k is the lecturer who offers p_j, we

- define $\theta_{i,j} = 1 - \sum_{p_{j'} \in S_{i,j}} x_{i,j'} \implies \theta_{2,1} = 1 - x_{2,1} = 1$.
- create $\alpha_j \in \{0,1\}$, enforce $c_j \alpha_j \geq c_j - \sum_{i'=1}^{n_1} x_{i',j} \implies \alpha_1 = 1$.

Blocking pair constraints

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
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: $d_1 = 2,\ d_2 = 1$.

For each (s_i, p_j), where l_k is the lecturer who offers p_j, we

- define $\theta_{i,j} = 1 - \sum_{p_{j'} \in S_{i,j}} x_{i,j'} \implies \theta_{2,1} = 1 - x_{2,1} = 1$.
- create $\alpha_j \in \{0, 1\}$, enforce $c_j \alpha_j \geq c_j - \sum_{i'=1}^{n_1} x_{i',j} \implies \alpha_1 = 1$.
- define $\gamma_{i,j,k} = \sum_{p_{j'} \in T_{k,j}} x_{i,j'}$;

Blocking pair constraints

Students' preferences		Lecturers' preferences
s_1: p_3 p_2 p_1		l_1: p_1 p_2
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: $d_1 = 2,\ d_2 = 1$.

For each (s_i, p_j), where l_k is the lecturer who offers p_j, we

- define $\theta_{i,j} = 1 - \sum_{p_{j'} \in S_{i,j}} x_{i,j'} \implies \theta_{2,1} = 1 - x_{2,1} = 1$.
- create $\alpha_j \in \{0,1\}$, enforce $c_j \alpha_j \geq c_j - \sum_{i'=1}^{n_1} x_{i',j} \implies \alpha_1 = 1$.
- define $\gamma_{i,j,k} = \sum_{p_{j'} \in T_{k,j}} x_{i,j'}$; $\quad T_{1,1} = \{p_2\}$

Blocking pair constraints

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
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: $d_1 = 2,\ d_2 = 1$.

For each (s_i, p_j), where l_k is the lecturer who offers p_j, we

- define $\theta_{i,j} = 1 - \sum_{p_{j'} \in S_{i,j}} x_{i,j'} \implies \theta_{2,1} = 1 - x_{2,1} = 1$.
- create $\alpha_j \in \{0,1\}$, enforce $c_j \alpha_j \geq c_j - \sum_{i'=1}^{n_1} x_{i',j} \implies \alpha_1 = 1$.
- define $\gamma_{i,j,k} = \sum_{p_{j'} \in T_{k,j}} x_{i,j'}$; $\quad T_{1,1} = \{p_2\} \implies \gamma_{2,1,1} = x_{2,2} = 1$.

Blocking pair constraints

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
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: $d_1 = 2,\ d_2 = 1$.

For each (s_i, p_j), where l_k is the lecturer who offers p_j, we

- define $\theta_{i,j} = 1 - \sum_{p_{j'} \in S_{i,j}} x_{i,j'} \implies \theta_{2,1} = 1 - x_{2,1} = 1$.
- create $\alpha_j \in \{0, 1\}$, enforce $c_j \alpha_j \geq c_j - \sum_{i'=1}^{n_1} x_{i',j} \implies \alpha_1 = 1$.
- define $\gamma_{i,j,k} = \sum_{p_{j'} \in T_{k,j}} x_{i,j'}$; $T_{1,1} = \{p_2\} \implies \gamma_{2,1,1} = x_{2,2} = 1$.

(i) $\theta_{i,j} + \alpha_j + \gamma_{i,j,k} \leq 2$;

Blocking pair constraints

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
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: $d_1 = 2,\ d_2 = 1$.

For each (s_i, p_j), where l_k is the lecturer who offers p_j, we

- define $\theta_{i,j} = 1 - \sum_{p_{j'} \in S_{i,j}} x_{i,j'} \implies \theta_{2,1} = 1 - x_{2,1} = 1$.
- create $\alpha_j \in \{0,1\}$, enforce $c_j \alpha_j \geq c_j - \sum_{i'=1}^{n_1} x_{i',j} \implies \alpha_1 = 1$.
- define $\gamma_{i,j,k} = \sum_{p_{j'} \in T_{k,j}} x_{i,j'}$; $\quad T_{1,1} = \{p_2\} \implies \gamma_{2,1,1} = x_{2,2} = 1$.

(i) $\theta_{i,j} + \alpha_j + \gamma_{i,j,k} \leq 2$; (ii) $\theta_{i,j} + \alpha_j + (1 - \beta_{i,k}) + \delta_k \leq 3$;

Blocking pair constraints

Students' preferences	Lecturers' preferences
s_1: p_3 p_2 p_1	l_1: p_1 p_2
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: $d_1 = 2,\ d_2 = 1$.

For each (s_i, p_j), where l_k is the lecturer who offers p_j, we

- define $\theta_{i,j} = 1 - \sum_{p_{j'} \in S_{i,j}} x_{i,j'} \implies \theta_{2,1} = 1 - x_{2,1} = 1$.
- create $\alpha_j \in \{0,1\}$, enforce $c_j \alpha_j \geq c_j - \sum_{i'=1}^{n_1} x_{i',j} \implies \alpha_1 = 1$.
- define $\gamma_{i,j,k} = \sum_{p_{j'} \in T_{k,j}} x_{i,j'}$; $T_{1,1} = \{p_2\} \implies \gamma_{2,1,1} = x_{2,2} = 1$.

(i) $\theta_{i,j} + \alpha_j + \gamma_{i,j,k} \leq 2$; (ii) $\theta_{i,j} + \alpha_j + (1 - \beta_{i,k}) + \delta_k \leq 3$;
(iii) $\theta_{i,j} + \alpha_j + (1 - \beta_{i,k}) + \eta_{j,k} \leq 3$.

Coalition constraints

Students' preferences	Lecturers' preferences
s_1: p_3 p_2 p_1	l_1: p_1 p_2
s_2: p_1 p_2	l_2: p_3
s_3: p_3	

Coalition constraints

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
s_2:	p_1	p_2		l_2:	p_3	
s_3:	p_3					

Envy graph

Coalition constraints

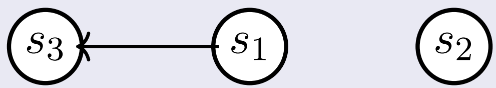
Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
s_2:	p_1	p_2		l_2:	p_3	
s_3:	p_3					

Envy graph

Coalition constraints

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
s_2:	p_1	p_2		l_2:	p_3	
s_3:	p_3					

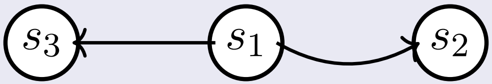
Envy graph



Coalition constraints

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
s_2:	p_1	p_2		l_2:	p_3	
s_3:	p_3					

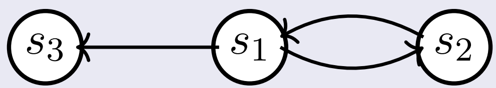
Envy graph



Coalition constraints

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
s_2:	p_1	p_2		l_2:	p_3	
s_3:	p_3					

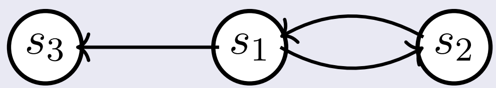
Envy graph



Coalition constraints

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
s_2:	p_1	p_2		l_2:	p_3	
s_3:	p_3					

Envy graph

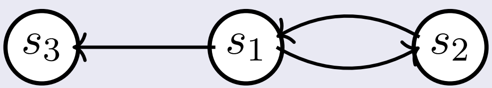


- admits topological ordering $\implies$ it is acyclic $\implies$ no coalition.

Coalition constraints

Students' preferences	Lecturers' preferences
s_1: p_3 p_2 p_1	l_1: p_1 p_2
s_2: p_1 p_2	l_2: p_3
s_3: p_3	

Envy graph

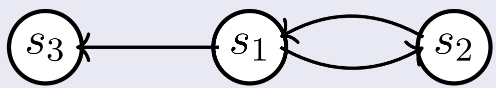


- admits topological ordering $\implies$ it is acyclic $\implies$ no coalition.
- For each $(s_i, s_{i'})$, if s_i envies $s_{i'}$, create $e_{i,i'} \in \{0, 1\}$ and enforce
  - $e_{i,i'} + 1 \geq x_{i,j} + x_{i',j'} \qquad i \neq i'$

Coalition constraints

Students' preferences				Lecturers' preferences		
s_1:	p_3	p_2	p_1	l_1:	p_1	p_2
s_2:	p_1	p_2		l_2:	p_3	
s_3:	p_3					

Envy graph



- admits topological ordering $\implies$ it is acyclic $\implies$ no coalition.
- For each $(s_i, s_{i'})$, if s_i envies $s_{i'}$, create $e_{i,i'} \in \{0,1\}$ and enforce
  - $e_{i,i'} + 1 \geq x_{i,j} + x_{i',j'} \qquad i \neq i'$
- to hold the label of each vertex in the topological ordering, create an integer-valued variable v_i and enforce
  - $v_i < v_{i'} + n_1(1 - e_{i,i'}) \qquad n_1$ – number of students.

Objective function

Objective function

- summation of all the $x_{i,j}$ binary variables

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

Objective function

- summation of all the $x_{i,j}$ binary variables

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

- it seeks to maximise the number of students assigned to projects

Objective function

- summation of all the $x_{i,j}$ binary variables

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

- it seeks to maximise the number of students assigned to projects

Theorem

Given an instance I of SPA-P, there exists an IP formulation J of I such that an optimal solution in J corresponds to a maximum stable matching in I, and vice-versa.

Implementation and Experimental Setup

Implementation and Experimental Setup

- IP model was implemented using the Gurobi optimisation solver
    - www.gurobi.com

Implementation and Experimental Setup

- IP model was implemented using the Gurobi optimisation solver
  - www.gurobi.com
- to investigate how the solution produced by the approximation algorithms compares to the optimal solution obtained from the IP model, with respect to the size of the stable matchings constructed

Implementation and Experimental Setup

- IP model was implemented using the Gurobi optimisation solver
  - www.gurobi.com
- to investigate how the solution produced by the approximation algorithms compares to the optimal solution obtained from the IP model, with respect to the size of the stable matchings constructed
- IP solver on instance size involving 1000 students
  - with the coalition constraints (63.50 seconds)
  - without the coalition constraints (2.61 seconds)

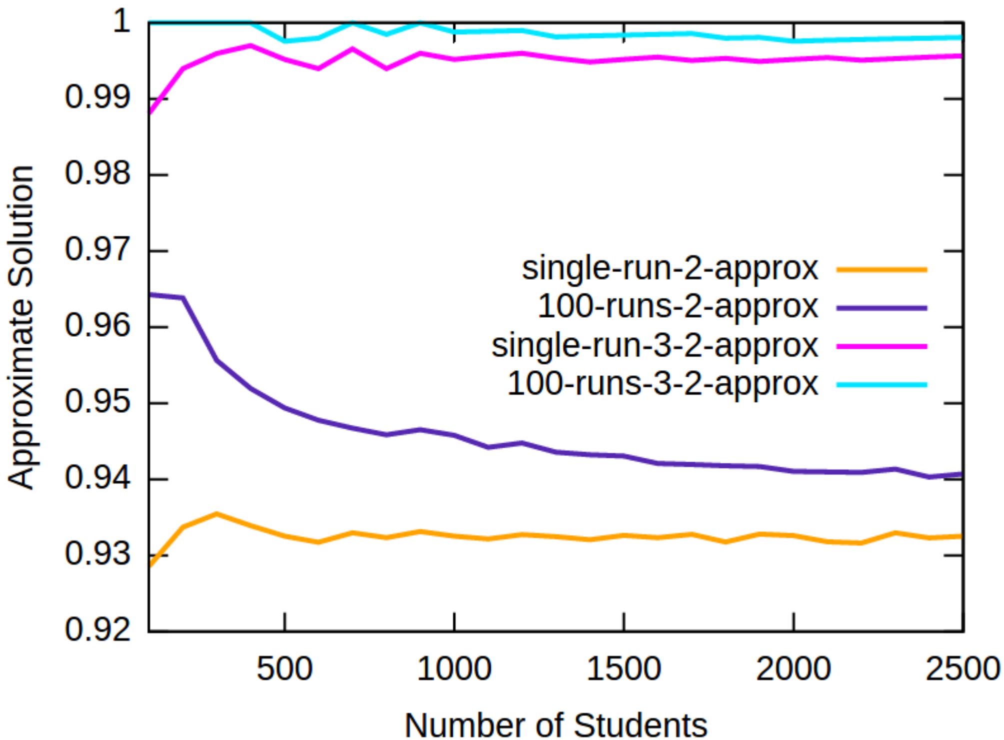
Implementation and Experimental Setup

- IP model was implemented using the Gurobi optimisation solver
  - www.gurobi.com
- to investigate how the solution produced by the approximation algorithms compares to the optimal solution obtained from the IP model, with respect to the size of the stable matchings constructed
- IP solver on instance size involving 1000 students
  - with the coalition constraints (63.50 seconds)
  - without the coalition constraints (2.61 seconds)
- size of a maximum stable matching = size of a matching that admits no blocking pair, but potentially admits a coalition

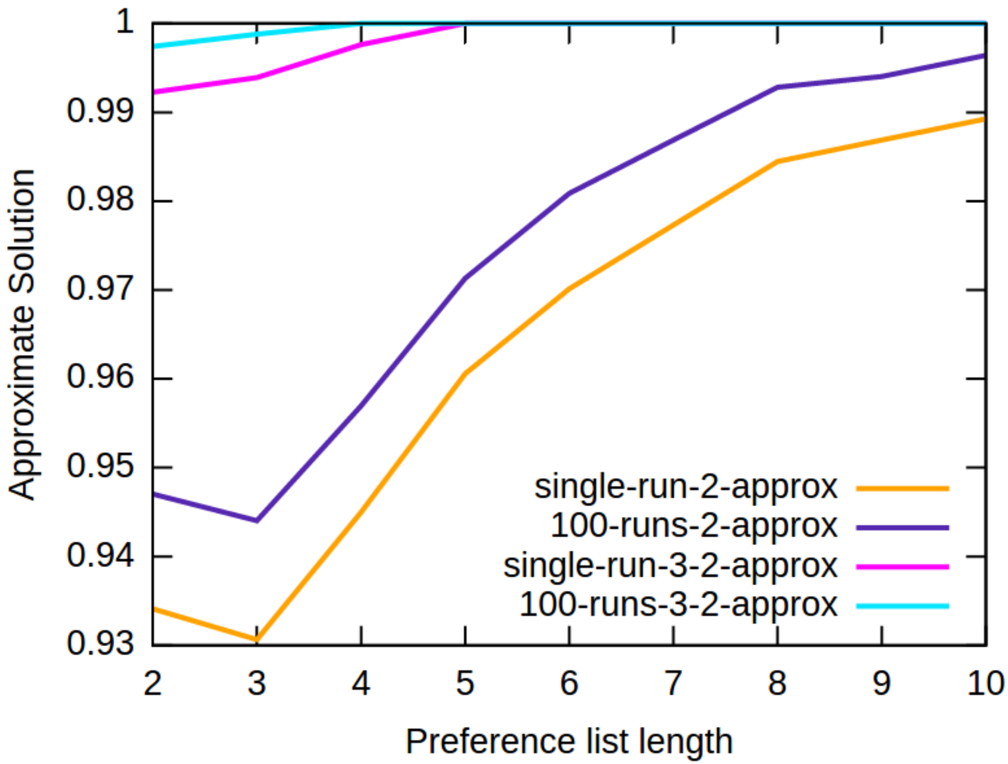
Implementation and Experimental Setup

- IP model was implemented using the Gurobi optimisation solver
  - www.gurobi.com
- to investigate how the solution produced by the approximation algorithms compares to the optimal solution obtained from the IP model, with respect to the size of the stable matchings constructed
- IP solver on instance size involving 1000 students
  - with the coalition constraints (63.50 seconds)
  - without the coalition constraints (2.61 seconds)
- size of a maximum stable matching = size of a matching that admits no blocking pair, but potentially admits a coalition
- for the purpose of this experiment, we removed the coalition constraints from our IP solver

Experimental results: Randomly-generated SPA-P instances



Experimental results: Randomly-generated SPA-P instances



Experimental results: SPA-P instances derived from real datasets

Experimental results: SPA-P instances derived from real datasets

- actual student preference data from previous runs of project allocation in the School of Computing Science, University of Glasgow; lecturer preference data was derived from this information

Experimental results: SPA-P instances derived from real datasets

- actual student preference data from previous runs of project allocation in the School of Computing Science, University of Glasgow; lecturer preference data was derived from this information

					Random					Most popular					Least popular				
Year	n_1	n_2	n_3	l	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

Table 1: A, B, C, D and E denotes the solution obtained from the IP model, 100 runs of $\frac{3}{2}$-approximation algorithm, single run of $\frac{3}{2}$-approximation algorithm, 100 runs of 2-approximation algorithm, and single run of 2-approximation algorithm respectively. Also, n_1, n_2, n_3 and l is number of students, number of projects, number of lecturers and length of the students' preference lists respectively.

Discussions and Conclusions

Discussions and Conclusions

- the approximation algorithms outperform the expected bound
- the $\frac{3}{2}$-approximation algorithm finds stable matchings that are very close in size to optimal, even on a single run

Discussions and Conclusions

- the approximation algorithms outperform the expected bound
- the $\frac{3}{2}$-approximation algorithm finds stable matchings that are very close in size to optimal, even on a single run
- IP solver on instance size involving $10,000$ students (100 seconds)

Discussions and Conclusions

- the approximation algorithms outperform the expected bound
- the $\frac{3}{2}$-approximation algorithm finds stable matchings that are very close in size to optimal, even on a single run
- IP solver on instance size involving $10,000$ students (100 seconds)
- IP model can be employed in practice

Discussions and Conclusions

- the approximation algorithms outperform the expected bound
- the $\frac{3}{2}$-approximation algorithm finds stable matchings that are very close in size to optimal, even on a single run
- IP solver on instance size involving $10,000$ students (100 seconds)
- IP model can be employed in practice
- potential coalitions can subsequently be dealt with in polynomial-time

Future work

Interesting directions..

Future work

Interesting directions..

- Approximation algorithm with improved bounds?

Future work

Interesting directions..

- Approximation algorithm with improved bounds?
- Fixed-Parameter Tractable (FPT) algorithm for MAX-SPA-P?

Future work

Interesting directions..

- Approximation algorithm with improved bounds?
- Fixed-Parameter Tractable (FPT) algorithm for MAX-SPA-P?
  - each project and lecturer has capacity 1

Future work

Interesting directions..

- Approximation algorithm with improved bounds?
- Fixed-Parameter Tractable (FPT) algorithm for MAX-SPA-P?
  - each project and lecturer has capacity 1 ✗

Future work

Interesting directions..

- Approximation algorithm with improved bounds?
- Fixed-Parameter Tractable (FPT) algorithm for MAX-SPA-P?
    - each project and lecturer has capacity 1 ✗
    - all preference lists are of bounded length

Future work

Interesting directions..

- Approximation algorithm with improved bounds?
- Fixed-Parameter Tractable (FPT) algorithm for MAX-SPA-P?
  - each project and lecturer has capacity 1 ✗
  - all preference lists are of bounded length ✗

Future work

Interesting directions..

- Approximation algorithm with improved bounds?
- Fixed-Parameter Tractable (FPT) algorithm for MAX-SPA-P?
  - each project and lecturer has capacity 1 ✗
  - all preference lists are of bounded length ✗
  - what if there is a constant number of lecturer?

Future work

Interesting directions..

- Approximation algorithm with improved bounds?
- Fixed-Parameter Tractable (FPT) algorithm for MAX-SPA-P?
    - each project and lecturer has capacity 1 ✗
    - all preference lists are of bounded length ✗
    - what if there is a constant number of lecturer?
    - might be solvable in polynomial-time with one lecturer?

Future work

Interesting directions..

- Approximation algorithm with improved bounds?
- Fixed-Parameter Tractable (FPT) algorithm for MAX-SPA-P?
  - each project and lecturer has capacity 1 ✗
  - all preference lists are of bounded length ✗
  - what if there is a constant number of lecturer?
  - might be solvable in polynomial-time with one lecturer?
  - remains hard to solve with two lecturers, even if each project has capacity 1

Future work

Interesting directions..

- Approximation algorithm with improved bounds?
- Fixed-Parameter Tractable (FPT) algorithm for MAX-SPA-P?
  - each project and lecturer has capacity 1 ✗
  - all preference lists are of bounded length ✗
  - what if there is a constant number of lecturer?
  - might be solvable in polynomial-time with one lecturer?
  - remains hard to solve with two lecturers, even if each project has capacity 1 ✓

Future work

Interesting directions..

- Approximation algorithm with improved bounds?
- Fixed-Parameter Tractable (FPT) algorithm for MAX-SPA-P?
  - each project and lecturer has capacity 1 ✗
  - all preference lists are of bounded length ✗
  - what if there is a constant number of lecturer? ✗
  - might be solvable in polynomial-time with one lecturer?
  - remains hard to solve with two lecturers, even if each project has capacity 1 ✓

Future work

Interesting directions..

- Approximation algorithm with improved bounds?
- Fixed-Parameter Tractable (FPT) algorithm for MAX-SPA-P?
  - each project and lecturer has capacity 1 ✗
  - all preference lists are of bounded length ✗
  - what if there is a constant number of lecturer? ✗
  - might be solvable in polynomial-time with one lecturer?
  - remains hard to solve with two lecturers, even if each project has capacity 1 ✓
  - more parameters yet to be explored..

Thank you for your attention

David Manlove[1], Duncan Milne and Sofiat Olaosebikan[2]. An Integer Programming Approach to the Student-Project Allocation Problem with Preferences over Projects. *To appear in proceedings of ISCO 2018: the 5th International Symposium on Combinatorial Optimisation, Lecture Notes in Computer Science, Springer, 2018.*

Corresponding author: Sofiat Olaosebikan
Website: www.dcs.gla.ac.uk/~sofiat
Email: s.olaosebikan.1@research.gla.ac.uk

[1] Supported by grant EP/P028306/1 from the Engineering and Physical Sciences Research Council.
[2] Supported by a College of Science and Engineering Scholarship, University of Glasgow.