Hard Variants of the Student-Project Allocation Problem

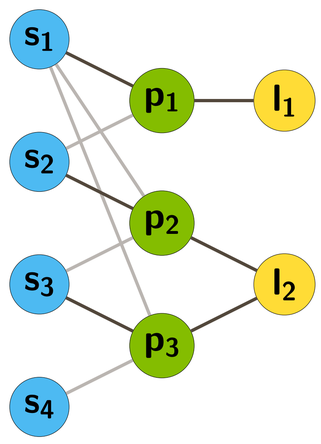
Frances Cooper and David Manlove, University of Glasgow, Scotland

We have a **matching problem** when we wish to assign one set of agents to another set of agents. Examples include assigning **children to schools**, **students to projects** and **kidney transplant patients to kidney donors**. Here we consider the **Student-Project Allocation problem** and describe how to cope with NP-hard variants.

The Student-Project Allocation Problem

The Student-Project Allocation problem with lecturer preferences over Students with Ties (SPA-ST)

- A set of students, projects and lecturers
- Each project is offered by a unique lecturer
- Students have preferences over projects, whilst lecturers have preferences over students
- Projects and lecturers have upper quotas
- Ties are allowed in lecturer (and student) preference lists



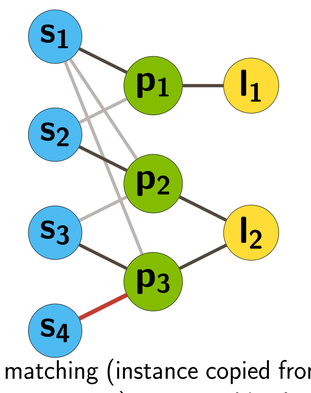
Example matching instance can either be viewed as a graph (above) or using preference lists (matching shown in bold):

s_1: ($\mathbf{p_1}$ p_2) p_3	p_1: UQ: 2	l_1: ($\mathbf{s_1}$ s_2)	UQ: 2
s_2: $\mathbf{p_2}$ p_1	p_2: UQ: 1	l_2: s_4 ($\mathbf{s_3}$ s_1) $\mathbf{s_2}$	UQ: 2
s_3: p_2 $\mathbf{p_3}$	p_3: UQ: 2		
s_4: p_3			

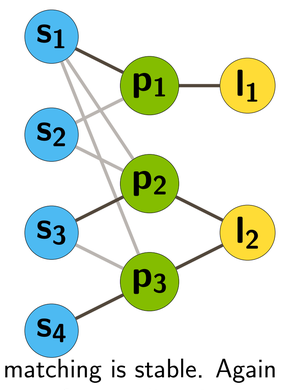
An extension to this problem allows lecturers to have **targets** which indicate the preferred number of allocations.

Optimisations

- A **stable matching** in SPA-ST is an assignment of students to projects such that upper quotas are respected and there is no student-project pair (s_i,p_j) where s_i and l_k, the lecturer offering p_j, have an incentive to deviate from the assignments (if any) and form a pairing.



This matching (instance copied from the previous section) is not stable. Looking at the preference lists in the previous section, student s_4 would prefer to be assigned to project p_3 and lecturer l_2 would also prefer this change.



This matching is stable. Again looking at the preference lists in the previous section, students s_1, s_3 and s_4 all have a first choice project. s_2 would like to be assigned either p_1 or p_2 but neither of the lecturers would prefer this change.

- **maximum size** - maximum number of students assigned
- Extension: **load balancing** - variety of comparisons between the number of lecturer allocations and targets

Important Results

- Every instance of SPA-ST admits a stable matching [1]
- A stable matching can be found in linear time [1]
- Stable matchings can have different sizes and finding a maximum sized stable matching in an instance of SPA-ST is NP-hard [2]

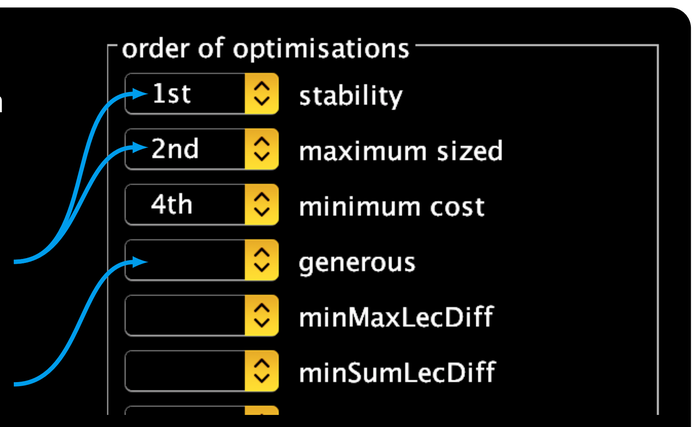
Finding Optimal Results with Integer Programming

Integer Programming (IP) is a computational technique that can deal with NP-hard optimisation problems. Finding a maximum stable matching in an instance of SPA-ST is NP-hard and so an **IP model** was developed.

- New integer inequalities and objective functions created for stability constraints and load balancing optimisations

Java Application

- Integer Program accessed by Java application
- Optimisations can be performed in any order



Generated Results

Input datasets were generated randomly and vary parameters such as the prevalence of ties in preference lists (Figure 1).

- 0%, 2.1% and 82% of instances timeout for 0.0, 0.1 and 0.2 ties probability respectively
- As tie probability increased, matching size and time taken to solve also increased
- Future work: bigger instances, longer timeouts

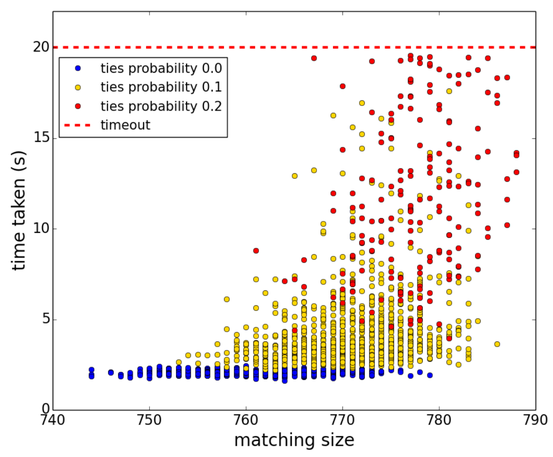


Figure 1: Preliminary results. Changes in time taken to solve instances versus matching size when varying preference list ties probability. In all cases there are 800 students, 350 projects, 200 lecturers, all lower quotas 0, all upper quotas 1000.

Real-World Results

In addition to generated data, the IP model has been used on **several real world scenarios** including student-project allocations for the University of Glasgow, the University of Edinburgh and the University of Leeds, and teacher-region allocations for TeachFirst. Each scenario had varying requirements but in several instances the IP model replaced a manual allocation process which was both time-consuming and unlikely to result in an optimal outcome.

University of Glasgow · UNIVERSITY OF LEEDS · The University of Edinburgh · TeachFirst

Finding Approximate Results with Polynomial Time Algorithms

Integer programming solvers use algorithms that run in exponential time in the worst case. Therefore, depending on priorities, it might be preferred that an **approximate** solution is found **quickly**.

Important Results

- For a simplified version of SPA-ST in which projects and lecturers are indistinguishable, it is possible to find a matching at least 2/3 the size of the maximum stable matching in linear time [3]

- A polynomial time **approximation algorithm** is currently being created that generalises the above result, and finds a stable matching at least 2/3 the size of the maximum stable matching for SPA-ST.

References

[1] D.J. Abraham, R.W. Irving, and D.F. Manlove. Two algorithms for the student-project allocation problem. Journal of Discrete Algorithms, 5:73-90, 2007.

[2] D.F. Manlove, R.W. Irving, K. Iwama, S. Miyazaki and Y. Morita. Hard variants of stable matchings. Theoretical Computer Science, 276(1-2):261-279, 2002.

[3] Z Király. Linear time local approximation algorithm for maximum stable marriage. Algorithms, 6:471-484, 2013.

Second year PhD student supervised by David Manlove and Kitty Meeks. Supported by EPSRC as DTA funding.