Integer Linear Programming and its Application in Carleton's CS Match

Batmend Batsaikhan, Andreas Miller, Mary Blanchard, Petrichor Park, Cecilia Ehrlichman, Hugh Shanno

Carleton College Computer Science Department

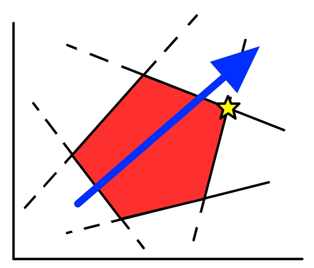
Carleton

1. Introduction

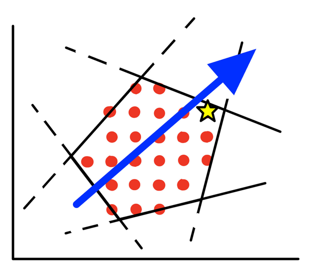
1. Understanding Integer Linear Programming

Solve for $x_1,\ x_2, \dots,\ x_N$

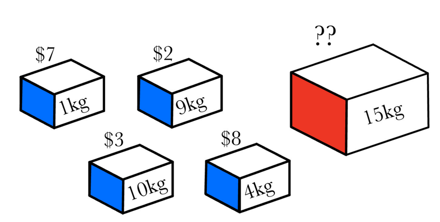
Objective Function: $\sum_{i=1}^{N} c_i x_i$
Constraints: $\sum_{i=1}^{N} a_{j,i} x_i \lesseqgtr b_i$



Integer Variables



2. Example: Knapsack Problem



Maximize the total value without exceeding the weight limit

Let $x_i = 1$ if the ith item is taken and $x_i = 0$ if otherwise.

Maximize $7x_1 + 2x_1 + 3x_3 + 8x_4$
While respecting $1x_1 + 9x_1 + 10x_1 + 4x_1 \le 15$
Where $0 \le x_i \le 1$

3. Carleton's CS Match

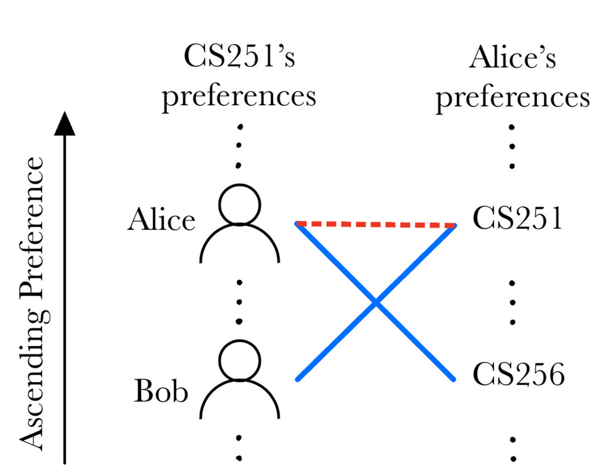
$\ge$ **300** students
~ **10** courses
Only **74%** first-choice matches
Currently uses the Gale Shapley Algorithm

4. Preferences and Stable Matching

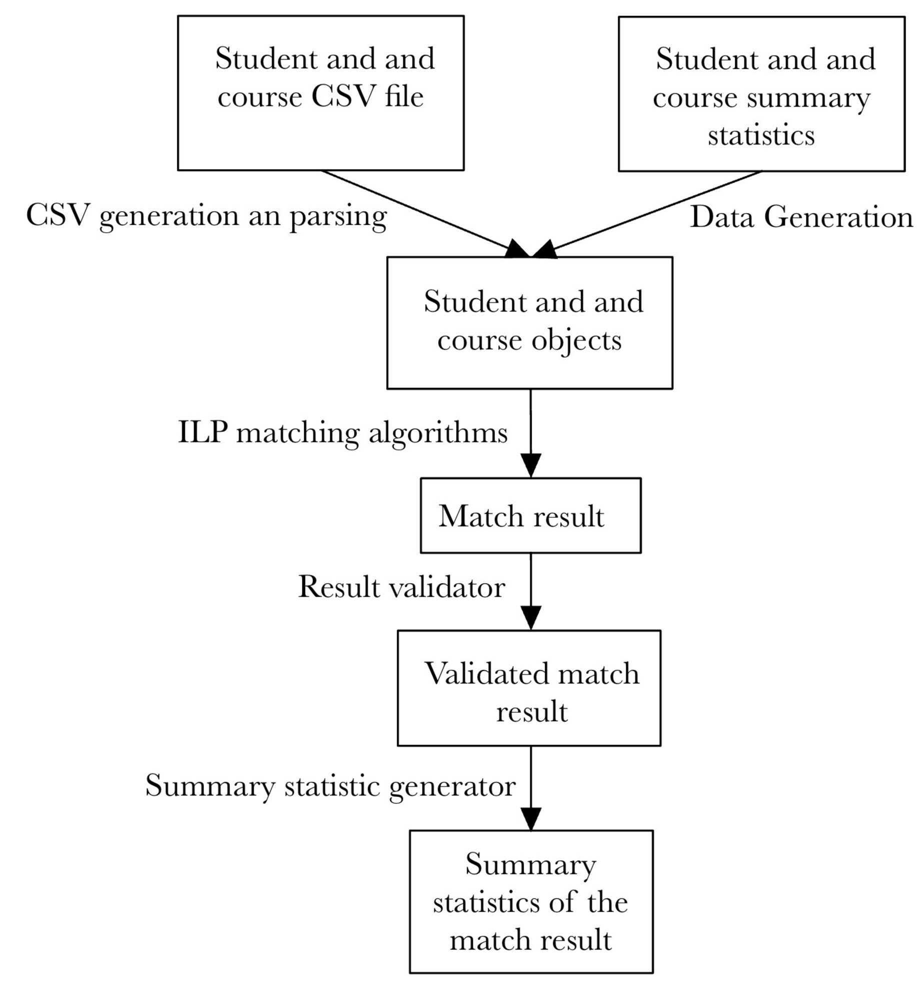
Student and Course preference Lists: Each student submits a preference list of courses

Courses rank students based on seniority, number of courses taken, etc.

Stable Matching: After a match, no unmatched pair wants each other more than what they are matched to



2. Work Structure



3. ILP formulations of the CS Match

For ith student and jth course, let $x_{ij} = \begin{cases} 0, & \textit{if } i \textit{ and } j \textit{ didn't match} \\ 1, & \textit{if } i \textit{ and } j \textit{ matched} \end{cases}$

1. Students get matched to at most one course

$$\sum_{j=1}^{C} x_{ij} \le 1$$

2. Courses don't exceed their capacity

$$\sum_{i=1}^{S} x_{ij} \le Capacity_j$$

3. "Stable Matching" constraint

For ith student and jth course
let S = $\{i' | i'$th student is not worse than ith student for jth course$\}$
let C = $\{j' | j'$th course is not worse than jth course for ith student$\}$

$$Capacity_j\Big(1 - \sum_{j' \in C} x_{ij'}\Big) \le \sum_{i' \in S} x_{i'j}$$

4. Objective functions

Hospital Resident Matching with Ties.

$$Maximize \sum_{i=1}^{C} \sum_{j=1}^{S} x_{ij}$$

Gale Shapley Emulation

$$Maximize \sum_{i=1}^{C} \sum_{j=1}^{S} x_{ij}\,(10000 - index(i,j))$$

Where $index(i,j)$ is the **order** of jth course for ith student

Weighted Matching

$$Maximize \sum_{i=1}^{C} \sum_{j=1}^{S} x_{ij}\,(10000 + weight(i,j))$$

Where weight(i,j) is the **weight** of jth course for the ith student

4. Specialization: Simplex Algorithm

Standard Form

Minimize $\sum_{i=1}^{N} c_i x_i$
With respect to $\sum_{i=1}^{N} a_{j,i} x_i = b_i$
Where $x_i \ge 0$

Tableau Form

x_1	x_2	x_3	x_4	x_5	f	RHS
2	2	0	1	2	0	7
1	1	1	3	0	0	6
1	2	3	1	0	0	6
2	2	−1	2	1	−1	0

Canonical Form

x_1	x_2	x_3	x_4	x_5	f	RHS
1	0	0	15/2	−1	0	15/2
0	1	0	−7	2	0	−4
0	0	1	5/2	−1	0	5/2
0	0	0	7/2	−2	−1	−9/2

Pivot

x_1	x_2	x_3	x_4	x_5	f	RHS
1	1/2	0	4	0	0	11/2
0	1/2	0	−7/2	1	0	−2
0	1/2	1	−1	0	0	1/2
0	1	0	−7/2	0	−1	−17/2

Repeat

x_1	x_2	x_3	x_4	x_5	f	RHS
1/4	1/8	0	1	0	0	11/8
7/8	15/16	0	0	1	0	45/16
1/4	5/8	1	0	0	0	15/8
7/8	23/16	0	0	0	−1	−59/16

Results $x_1 = x_2 = 0, x_3 = \frac{15}{8}, x_4 = \frac{11}{8}, x_5 = \frac{45}{16}, f = \frac{59}{16}$

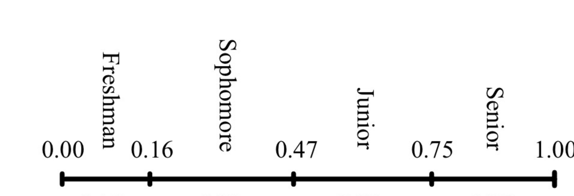
Finiteness of the Algorithm: Bland Rule: Out of equivalent choices (rows or columns), pick one that has the minimum index

5. Specialization: Data Generation

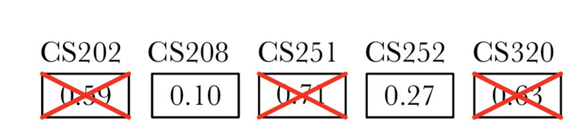
Accessed only summary statistics to respect the privacy of students

	# of total students	Mean # of classes wanted
Freshman	57	4.15
Sophomore	96	4.51
Junior	97	3.85
Senior	61	2.43

1. Generate Class Year
$c \sim U(0,1)$



2. Generate course weights based on class year
$w_i \sim U(0,1)$

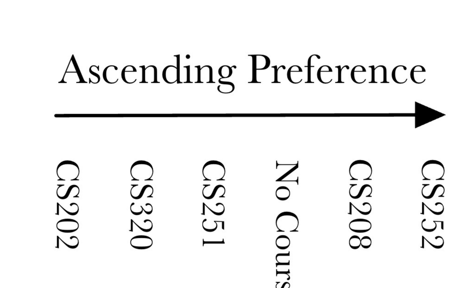


4.15 / 9 = 0.45 cutoff

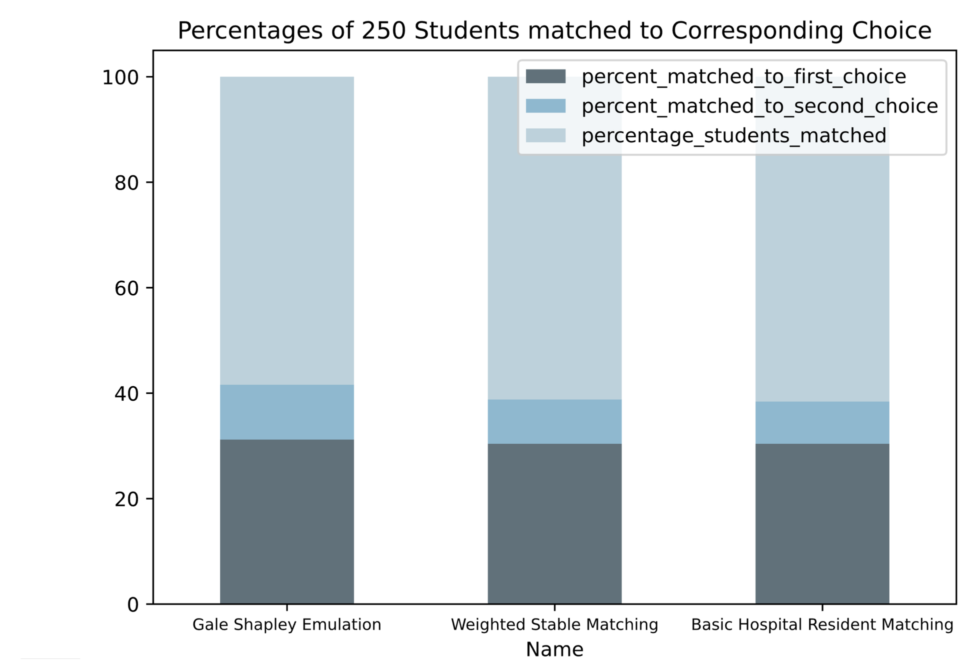
3.1 Generate normalized weights

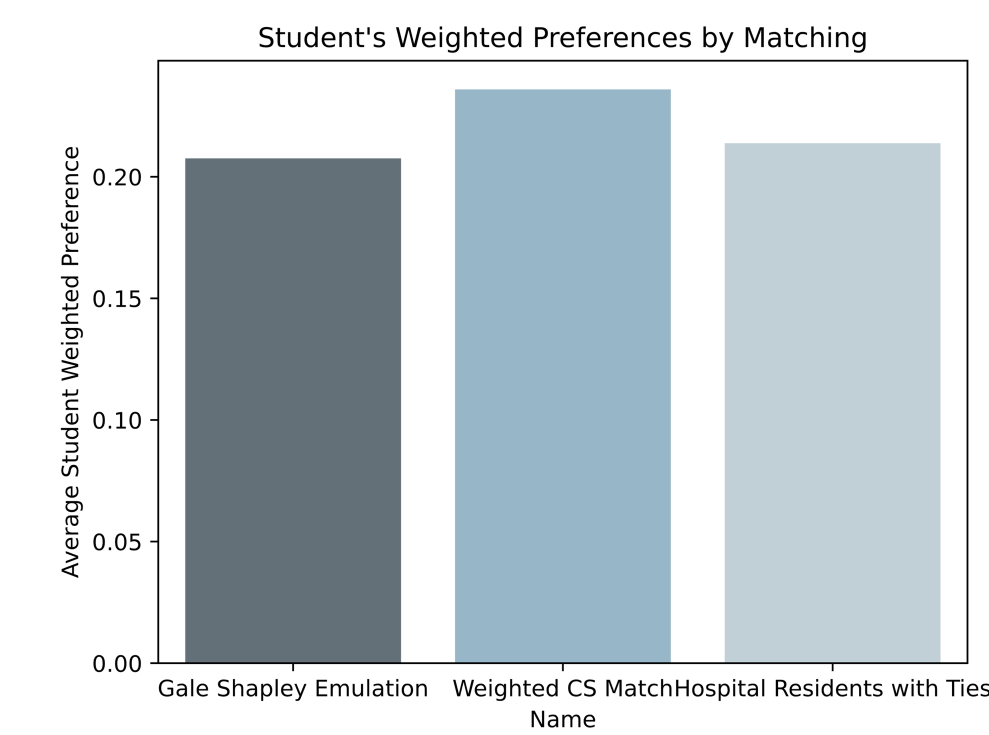
CS202	CS208	CS251	CS252	CS320
0.00	0.27	0.0	0.73	0.00

3.2 Generate preference list



6. Preliminary Results





7. Conclusion

Integer Linear Programming is a flexible technique that can model not only trivially linear problems but also various optimization problems with non-trivial structures.

Adjusting the objective function provides us with even greater flexibility in defining what is the most optimal, consequently influencing the resulting solutions of the ILP algorithm.

Initially, we had intended to run our ILP models against Carleton's CS Match algorithm using the same generated dataset. We had to forego this plan due to time constraints. It is important for the study to running such comparisons in the future.

8. Acknowledgements

I would like to express my gratitude to Professor Layla Oesper for her guidance and Professor David Musicant for sharing summary data. My deepest appreciation goes to my parents, brothers, and friends for their unwavering support throughout my academic journey.

9. References

PAN, P.-Q. (2023). *Linear Programming Computation* (2nd ed. 2023.). Springer Nature Singapore. https://doi.org/10.1007/978-981-19-0147-8

Delorme, M., García, S., Gondzio, J., Kalcsics, J., Manlove, D., & Pettersson, W. (2019). Mathematical models for stable matching problems with ties and incomplete lists. *European Journal of Operational Research, 277*(2), 426–441. https://doi.org/10.1016/j.ejor.2019.03.017