Applications of Integer Linear Programming (ILP): The CS Match (+ Ties and Stability)

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

INTRODUCTION

- ILP is a type of optimization problem
- ILP Formulations consist of three components:
  - Variables – must have integer solutions
  - Constraints – inequality or equation that is linear
  - Objective Function – either maximized or minimized
- Feasible Solution – any solution that fits the constraints
- Optimal Solution: Feasible Solution that produces the largest/smallest output for Objective Function

ILP Formulation Example

Variables: x_1, x_2

Constraints: $2x_1 + x_2 \leq 100$ (1)

$x_1 + x_2 \leq 80$ (2)

$x_1 \leq 40$ (3)

$x_1, x_2 \geq 0$

Objective (Maximize): $Z = 3x_1 + 2x_2$

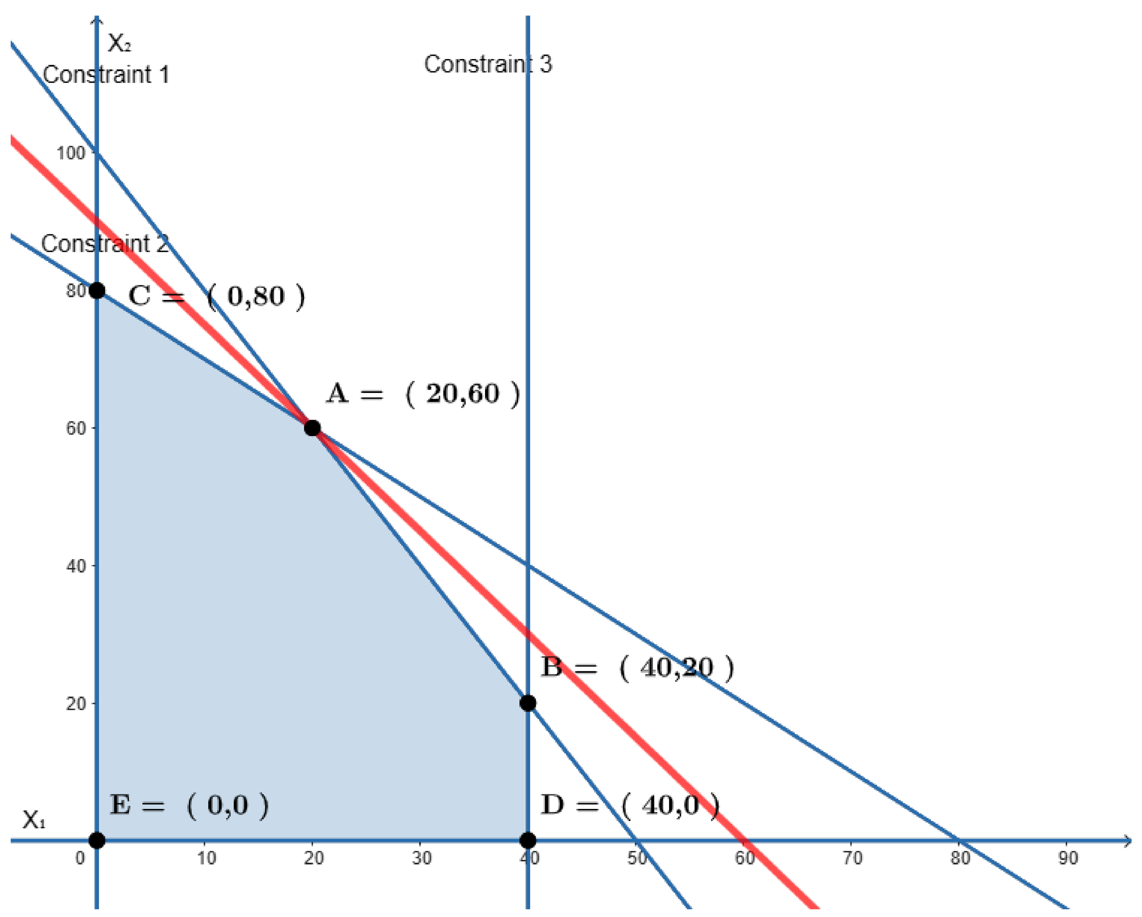

Figure 1

Optimal Solution: $x_1 = 20, x_2 = 60$

*Found at point **A***

ACKNOWLEDGEMENTS

Layla Oesper, for assisting as our Comps advisor

Dave Musicant, for supplying metrics for data generation

BACKGROUND

- CS Match aims to help as many students as possible to register for a CS class for a term,
- CS Match is a stable matching problem and currently solved with Gale-Shapley algorithm
  - Students list preferred courses and vice versa
- We sought to reformulate the stable matching problem as an ILP problem
- We sought to compare the matching results between the CS Match and the ILP
- We sought to compare strict ranking with weights and introduce tied preferences in the ILP

CODE STRUCTURE

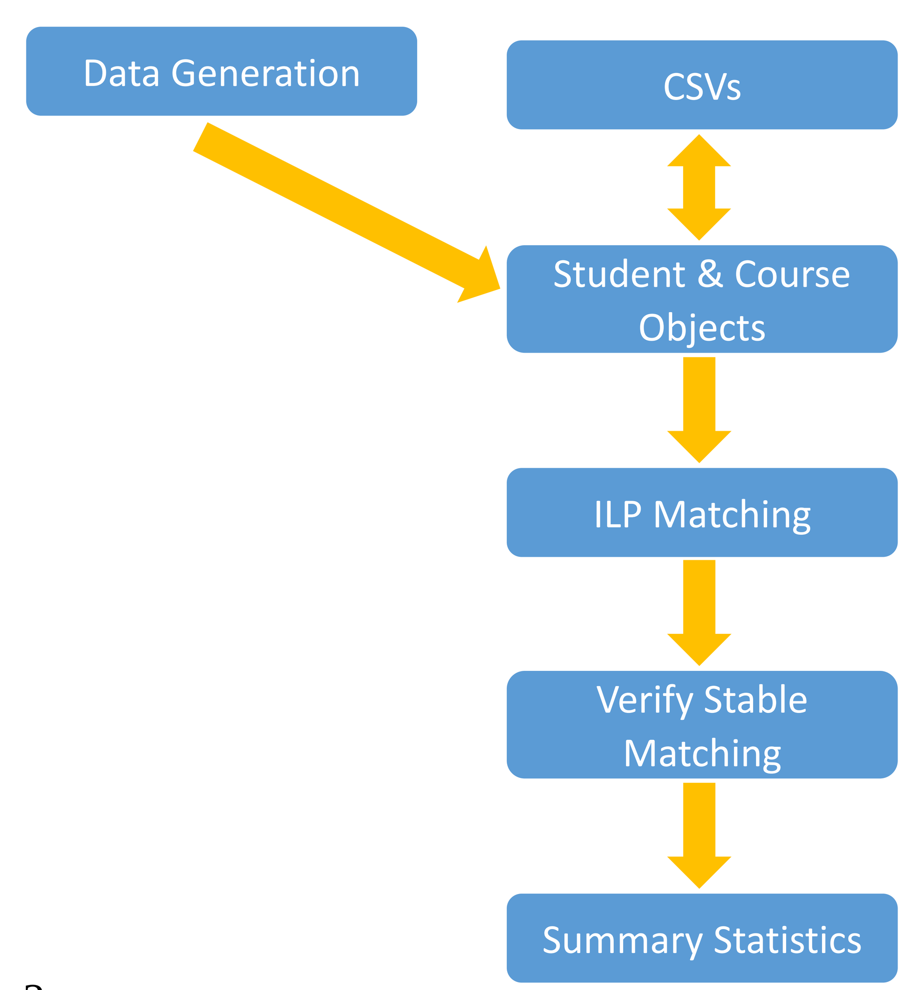

Figure 2

VALIDATING STABILITY

- Weakly stable – No blocking pair exists in a matching
- Blocking pair – an unpaired student *s* and course *c* in a matching *M* where *s* is either unassigned or strictly prefers *c* to their assigned course in M and *c* is either not full or strictly prefers *s* to the least preferable student assigned to it in *M*
- Super-stable – No blocking pair exists in a matching, where blocking pair is no longer strict (i.e. includes greater than <u>or equal</u> preferences)

ILP FORMULATION

<u>Notation</u>

$(1, \ldots, n_1)$ ~ set of all students

$(1, \ldots, n_2)$ ~ set of all courses

k_j ~ capacity of a course j

x_{ij} ~ binary variable for match between student i and course j

$C(i)$ ~ set of courses wanted by student i

$S(j)$ ~ set of students wanted by course j

$C_j^{\leq}(i)$ ~ set of courses that student i ranks at the same level or better than course j

$S_i^{\leq}(j)$ ~ set of students that course j ranks at the same level or better than student i

<u>Math</u>

Variables: x_{ij} $\quad i = 1, \ldots, n_1; j \in C(i)$

Constraints:

(1) $\sum_{j \in C(i)} x_{ij} \leq 1$ $\quad i = 1, \ldots, n_1$

(2) $\sum_{i \in S(j)} x_{ij} \leq k_j$ $\quad j = 1, \ldots, n_2$

(3) $k_j\left(1 - \sum_{q \in C_j^{\leq}(i)} x_{iq}\right) \leq \sum_{p \in S_i^{\leq}(j)} x_{pj}$ $\quad i = 1, \ldots, n_1;\ j \in C(i)$

(4) $x \in \{0,1\}$ $\quad i = 1, \ldots, n_1;\ j \in C(i)$

Objective Function:

(5) Max $\sum_{i=1}^{n_1} \sum_{j \in C(i)} x_{ij}$

<u>Intuition</u>

(1) Each student is matched with one course, if any

(2) Each course does not exceed its capacity

(3) Stability: if student i was not assigned to course j or any other course they rank at the same level or higher than j, then course j has filled its capacity with students it ranks at the same level or higher than i

(4) Match between student and course pair either exists $[x = 1]$ or doesn't $[x = 0]$

(5) Maximize the number of students matched to a course

RESULTS

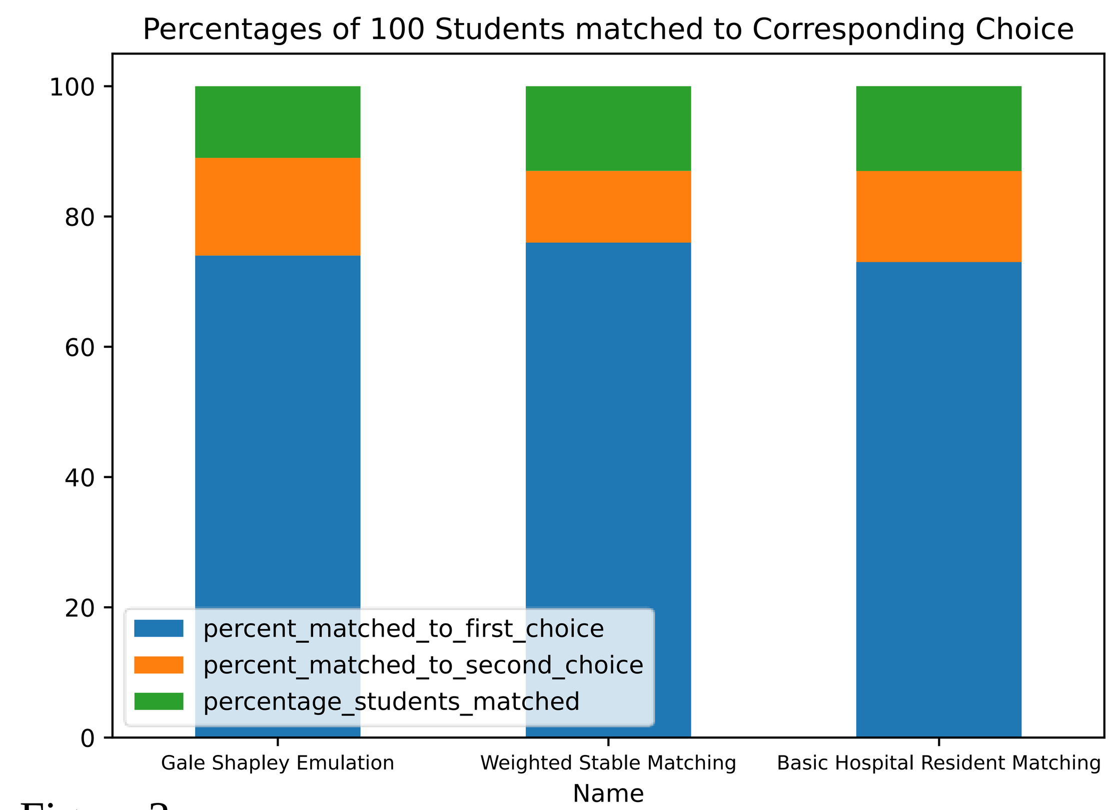

Figure 3

MATCHING WITH TIES

- Preferences located within the same tie are considered equal
- The order in which ties are broken can result in stable matches of different sizes
- Tie density $t_d (0 \leq t_d \leq 1)$ of the preference lists is the probability that some agent is tied to the agent next to it in a preference list
- As tie density increases, the average size of maximum stable matchings increases

Students' Preferences

$S_1: C_1\ C_3$

$S_2 : C_1$

$S_3 : C_1\ C_2$

$S_4 : C_3$

$S_5 : (C_2\ C_3)$

$S_6 : C_1\ C_3$

Courses' Capacities and Preferences

$C_1 : \{2\} : S_1\ S_2\ S_3\ S_4\ S_5\ S_6$ [CORE]

$C_2 : \{2\} : S_1\ S_2\ S_3\ S_4\ S_5\ S_6$ [CORE]

$C_3 : \{2\} : S_1\ S_2\ S_3\ S_6\ S_5\ S_4$ [ELECTIVE]

**entries in paratheses are tied*

Stable Matchings	Matching Size
$M_0 = \{(S_1, C_1), (S_2, C_1), (S_3, C_3), (S_5, C_2), (S_6, C_2)\}$	5
$M_1 = \{(S_1, C_1), (S_2, C_1), (S_3, C_3), (S_4, C_2), (S_5, C_3), (S_6, C_2)\}$	6

Figure 4

REFERENCES

- 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.
- Irving, Robert W., et al. "The Hospitals/Residents Problem With Ties." Lecture Notes in Computer Science, 2000, pp. 259–71. https://doi.org/10.1007/3-540-44985-x_24.
- Kwanashie, Augustine, and David F. Manlove. "An Integer Programming Approach to the Hospitals/Residents Problem With Ties." Operations Research Proceedings, 2014, pp. 263–69. https://doi.org/10.1007/978-3-319-07001-8_36.
- Winston, W. L. (1996). Operations research: Applications and Algorithms. Brooks/Cole.