Marco Calabretta
June 17, 2024
On the inadequacy of the UW Co-op matching process

The University of Waterloo's co-op matching algorithm has a problem. Likely it has many problems, but this discussion will focus on one important property that the matching process does not have, which I will call "**stability**". If a matching algorithm is **stable**, that means that, assuming all students and employers honestly rank each other by preference, a student is effectively stuck with his job outcome after the matching algorithm has finished (Note: this concept of stability was inspired by this video). To be more rigorous, suppose student s gets matched with job j, which they ranked n. Then, the matching algorithm is **stable** if and only if, for all $j_i \in \{j_1, j_2, ...j_{n-1}\}$ which they ranked higher, j_i is guaranteed to be matched with some student s_i, such that j_i ranked s_i before s. If this was not the case, there would be some j_i that s prefers over j that also prefers s over s_i, and there would be a theoretical potential for boththe student and the employer to sign a co-op contract with each other and break the existing matches that they made through WaterlooWorks.

I think this is an important property for any matching algorithm to have, because it seems like the best that can be done in terms of guaranteeing good matches for students and employers. Obviously no algorithm can guarantee that everyone gets the job or student that they wanted, but guaranteeing that they can't improve on their match is a close second.

Now, we will show that the WaterlooWorks matching algorithm is not **stable** and thus should not be used to match students to jobs. Here's the proof by counterexample:
Suppose there exist students A, B, C, D, and they've ranked openings E, F, G, H as follows:

Rank\Student	A	B	C	D
1	E	F	E	G
2	F	E	F	E
3	G	G	G	F
4	H	H	H	H

Also, the employers have ranked the students as follows:

Rank\Opening	E	F	G	H
1	A	B	A	A
2	B	A	B	B
3	C	C	C	C
4	D	D	D	D

Now we execute the current matching algorithm by first calculating the ranking sums between each student-job pair:

Opening\Student	A	B	C	D
E	2	4	4	6
F	4	2	5	7
G	4	5	6	5
H	5	6	7	8

Clearly, A and E get matched with each other, and so do B and F, since they both ranked each other first. That leaves us with this reduced table:

Opening\Student	C	D
G	6	5
H	7	8

Based on the lowest sum, we match C with G, and then are left with a D-H match. However, D prefers G over H, *and* G prefers D over C. In this example we've broken the requirements for stability, and therefore the WaterlooWorks match algorithm is not stable. **QED**

Since the current algorithm is not stable, we should try to replace it with an algorithm that is stable, or at least prove that such a task is impossible given the unique properties of the Waterloo co-op process.