Phuc Dang
Ptd328
EE360C
Lab 1

Lab 1 Report

1.a

Assume there is a hard worker that doesn't get a full time job. Thus, there exists a pair of (lazy worker – full time job) and a pair of (hardworking worker – part time job). However, that will be an unstable pair because the hardworking worker will prefer the full time job over his current part time job, and the full time job also prefers the hardworking worker over the lazy worker.

1.b

The Big O of permutation is N! where N is the number of jobs and workers, so the time complexity in big O of the brute force algorithm will be N! * big O of the isStableMatching function. My isStableMatching function has a big O of N^3. The reason why my isStableMatching has a run time of N^3 is because for each pair in the matching, it will illiterate through the loop to compare with all the pair in the matching. In order to compare two pairs, I need to search for the indexes in their preference list (indexOf function). The algorithm will terminate after going through all pairs (which is N). Thus the time complexity of the brute force algorithm will be N! *(N^3).

1.c

Initialize all j ∈ J and all w ∈ W to be free

While there exists a job who is free and has not been occupied by every w ∈ W

Do Choose such a job j

Let w be the highest ranked in j's preference list whom w has not been in pair with j

If w is free

Then (w, j) becomes a pair

Else w is paired with j'

If w prefers j' to j

Then j remains free

Else w prefers j to j'

(w, j) becomes a pair

j' becomes free

return the set of S pairs

1.d

We have already proved the correctness of the Gale-Shapley in class. I am using the lecture slides as a reference to answer this question.

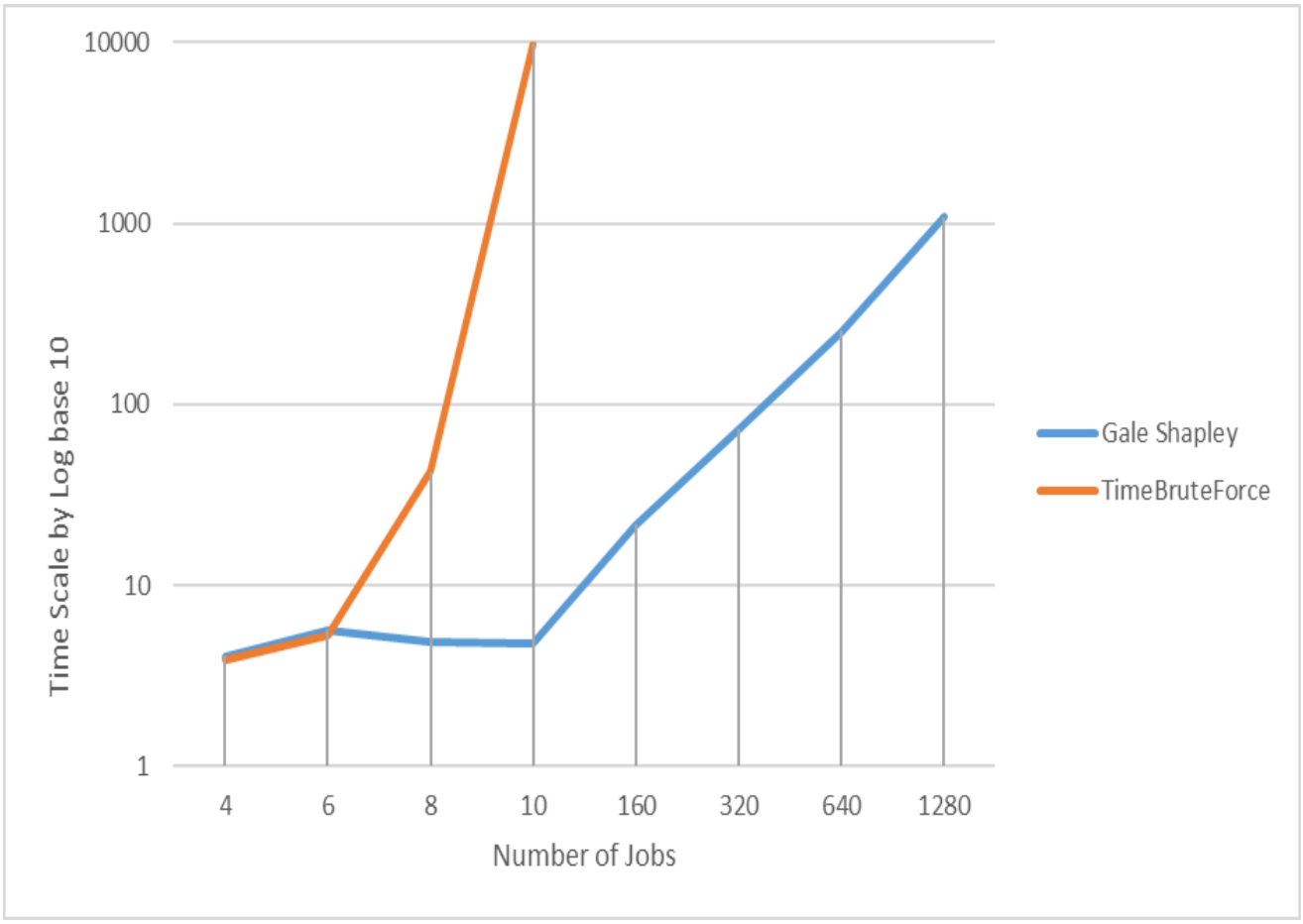
The algorithm will terminate at most N^2 iteration of the while loop. Each iteration consists of a job "proposed" to a worker it has never proposed to before. After each iteration of the while loop, the number of iteration increase by 1, so the total number of proposal is upper bounded by N^2. Thus termination is guaranteed.

This algorithm will return a stable matching. Assume there is an instability, then there exists two pairs (j, w) and (j', w') s.t j prefer w' to w and w' prefers j to j'. Thus, j will "proposed" to w' before w because w' is before w in his preference list. However, a worker can only reject a job if he receives a proposal from a job that he prefers. Thus, if a worker rejects a job then he prefers his final jobs over the rejected man, so w' prefer j' over j, which is a contradiction.

1.e

My algorithm is based on the Gale-Shapley algorithm. The workers represent the girls, and the jobs represents the boys. As mentioned above in 1.d, I have proved that the algorithm will terminate after at most N^2 iteration, so the time complexity of my algorithm is N^2.

1.f



4.a

No. Counter Example: Say n and m is 2. The preference lists are wo [1, 0] w1 [1, 0] j0[0, 0] and j1[0,0]. This will create this set of perfect matching: [(w0, j1), (w1, j0)] and [(w1, j1), (w0, j0)], and the rest are unstable. Both matchings have a weak instability because w0 and w1 both prefer j0 to j1, but j1 is indifferent between w0 and w1. Every perfect matching will have a weak instability.

4.b

Yes. If I use the algorithm that was used in the third part, it will be able to output a perfect matching with no strong instability. First, as we already prove above, it will always terminate after n^2 iteration and the result is always a perfect matching. The condition of strong instability is the same as instability in part 1, so we can say that this will also hold true. Thus, there always exists a perfect matching that have no strong instability.

4.c

My definition of stable matching is a perfect matching with no strong instability. The reason for this is because there is always a perfect matching with

Initialize all j ∈ J and all w ∈ W to be free

While there exists a job who is free and has not been occupied by every w ∈ W

Do Choose such a job j

Let w be the highest ranked in j's preference list whom w has not been in pair with j

If w is free

Then (w, j) becomes a pair

Else w is paired with j'

If w prefers j' to j

Then j remains free

Else w rank j strictly higher than j'

(w, j) becomes a pair

j' becomes free

return the set of S pairs