CSCI 570

Homework #1

Graded Problems

1) This is true. In unique stable matching only one stable matching is possible. If other perfect matching arrangements are tried, then there is not a stable matching because it will create an instability.

2) Two disjoint pairs i.e.

Pair 1	Pair 2
- a	- b
- c	- d

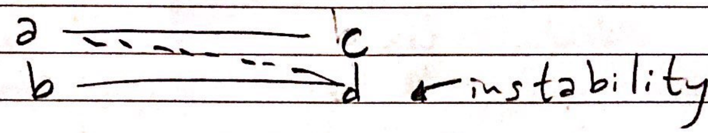
In this problem, a perfect matching will always exist because we can iterate over until two disjoint pairs are formed as our condition. Consider an execution of the G-S algorithm that returns a set of pairs S. The set S is a stable matching.

Proof: Since S is a perfect matching because we will not stop iterating until every student has proposed another student to live with him, for otherwise the While loop would not have exited, we will prove S is a stable matching. We will assume that there is an instability with respect to S an obtain a contradiction. Let's assume stable matching of the two disjoint pairs exist in the following configuration:

Pair 1: (a, c)
Pair 2: (b, d)

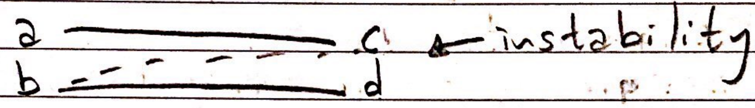
In the case where (a, c) and (b, d) creates an instability, then either

- a prefers d to c, and
- d prefers a to b



or

- c prefers b to a, and
- b prefers c to d



Let's choose the instability in the case where a and d want to be in the same pair for our contradiction.

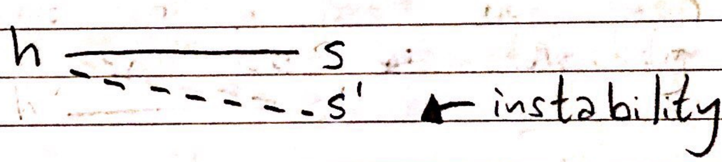
In the execution of the G-S algorithm that produces S a's last proposal was to because that is our final matching. However, we will like to know if a proposed to d at some earlier point in the execution. If a did not propose to d, then c must occur higher on a's preference list than d. As a result, this contradicts the assumption that a prefers d to c.

In contrast, if a had proposed to d, then a was reject by d in favor of some other student "x", whom d prefers to a. b is the final partner of d, so either "x" = b or d prefers its final partner b than "x", which should equal student c "x" = c. Either way, this contradicts our assumption that d prefers a to b, which creates an instability.

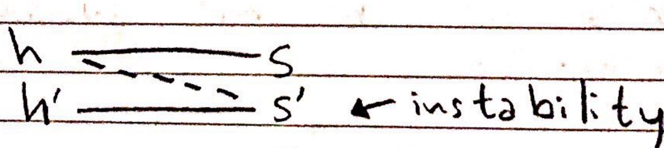
Therefore, it is shown that a stable matching always exist.

3) Two types of instabilities

- First type of instability: There are student s and s', and a hospital h, so that
  - s is assigned to h, and
  - s' is assigned to no hospital, and
  - h prefers s' to s



- Second type of instability: There are s and s', and hospitals h and h', so that
  - s is assigned to h, and
  - s' is assigned to h', and
  - h prefers s' to s, and
  - s' prefers h to h'



Exceptions: i) hospitals generally want more than one resident ii) there is a surplus of medical students

Gale - Shapley Algorithm:

$\in$ means is an element of. For instance, "Let $a \in A$" means "Let a be an element of A".

Therefore,

Initially all $h \in H$ and $s \in S$ are free
While there is an open slot in a hospital h and hasn't offered a job to every student s in the order of its preference list
  Choose such a hospital h
  Let s be the highest-ranked student in h's preference list to whom h has not offered a job yet

  If s does not have a job offer then
    s accepts the job and takes on an empty slot of needed residents
  Else s already accepted a job with h'
    If s prefers h' to h then
      h's job opening remains available
    Else s prefers h to h'
      s accepts the job offer from h
      h''s job becomes available for more students
    Endif
  Endif
Endwhile
Return the set S of jobs and its residents

Show that there is always a stable assignment of students to hospitals

From fact 1.6 from the book, Consider an execution of the G-S algorithm that returns a set of pairs S. The set S is a stable matching.

In this case, we could only have two types of instabilities:

For the first possible instability, we can assume h prefers s' to s and obtain a contradiction. In the execution of the algorithm that produced S, h's last job offer was to s. Now we ask: Did h offered a job to s' at some earlier point in this execution? If it did, then s' would work for h because s' is currently not assigned to a hospital. There is a contradiction. In addition, if h did not offer a job to s', then s must occur higher on h's preference list than s', contradicting our assumption that h prefers s' to s. Either way this contradicts our assumption that h prefers s' to s. Basically, s' was not offered a job.

It follows that with with the G-S algorithm, the first instability is not feasible.

For the second possible instability, we assume that there are two existing pairs: (h, s) and (h', s'). Furthermore, there is an instability created by h and s'. With the aforementioned, we will prove a contradiction.

In the execution of the algorithm that produced S, h's last job offer was to s if the algorithm is followed. Now we ask: Did h offered a job to s' at some earlier point in this execution? If it did not, then s must occur higher on h's preference list than s', contradicting our assumption that h prefers s' to s. If it did, then h was rejected by s' in favor of some other hospital h'', whom s' prefers to h. Since h' is the final employer of s', then h''= h' or s' prefers its final employer h' to h''. Regardless, this contradicts our assumption that s' prefers h to h'.

Finally, it follows that S is a stable matching. Thus, there is always a stable assignment of students to hospitals.

4) -Let Almazo Wider be denoted as A.
-Let Nelly Oleson be denoted as N.
-Let Laura Ingles be denoted as L.
-Let men be denoted as m, m', m'', and so forth.
-Let women be denoted as w, w', w'', and so forth.

When A swaps N and L on his preference list, an instability will be created if and only L prefers A to her current partner m. If that is the case, then m will propose to another woman and N will be waiting for a man to propose to her.

At the beginning of this algorithm, only A has swapped N and L on his preference list

If A is on a higher rank than L's current parther, then L gets engaged with A and L's former partner, m, is now free. Also, N is free.

While there is a man m who is free and has not proposed to every woman.

Choose such a man m

Let w be the highest-ranked woman in m's preference list to whom m has not yet proposed.

If w is free then
(m, w) become engaged

Else w is currently engaged to m'

If w prefers m' to m then
m remains free

Else w prefers **m** to m'
(m, w) become engaged
m' becomes free

Endif

Endif

Endwhile

Else A is not on a higher rank than L's current partner. Thus, A remains engaged to N

Endif

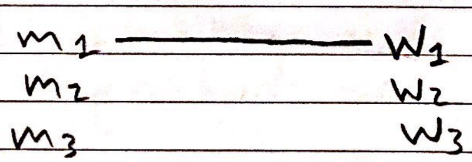
Return the set S of engaged pairs

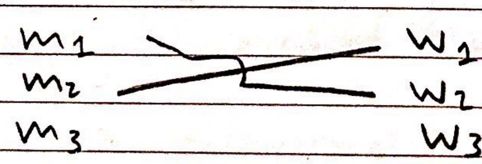
Practice Problems

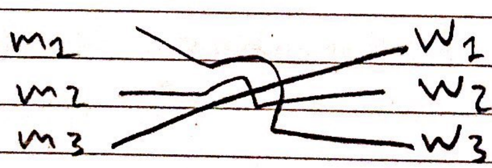
1) Chapter 1, Exercise 1

This is false. Here is a counterexample:

m_1	m_2	m_3	w_1	w_2	w_3
w_1	w_1	w_2	m_3	m_2	m_1
w_2	w_2	w_1	m_2	m_1	m_2
w_3	w_3	w_3	m_1	m_3	m_3







Pairs: (m_1, w_3), (m_2, w_2), (m_3, w_1)

2) Chapter 1, Exercise 2

This is true. If m is the best valid partner of w and it is ranked first on her preference list and viceversa. Then (m, w) ∈ S because m will always propose to w first and w will not break her engagement with m for another man.

3) Chapter 1, Exercise 3

Initially the TV shows of Network A and B are not in order. Therefore, they will not become a stable pair (S, T). Thus, we will sort them in descending order

While the schedules S and T are not in descending order

If the following element in the schedule is smaller than the current index swap them. Then, compare the values of the rest of the list with the recently swapped value by repeating this process recursively.

Else do not swap the elements and move to the following index on next iteration

Endif

Endwhile
Return a stable pair of schedules