Assignment 1

CSCI 270

Paul Kim | pbkim@usc.edu | 1723717002

Question 1

In this problem, we will consider what happens in stable matching when we are guaranteed additional structure on the preferences.

1. Suppose that there is a ”most popular” man, whom all women rank first. Describe who this man must be matched with in every stable matching, and explain why.
2. Use your insight from the previous part to analyze the case when all women have identical rankings. Prove that in this case, there is a unique stable matching, and describe it.
3. For the case of the previous problem, describe a simpler algorithm than Gale-Shapley which never has to break any engagements, and computes the unique stable matching. Prove that your algorithm outputs a stable matching. (Your proof may heavily draw on the previous sub problems.)

Answer:

**Note: A stable matching involves two types of entities (man (M) and woman (W)) in which, each entity has a certain preference over all entities of other types, and no pair (M,W) who were not assigned to each other would prefer each other over their assigned partner.*

1. Supposed that there is a ”most popular” man that every women ranks as their first preference, then it should be so that this man will always be matched to ”his” most popular women: in other words, the man’s first woman preference. The reason is because regardless if the man or woman makes the first move, the stability definition states that no pair who were not assigned to each other would prefer each other over their assigned partner. Therefore, it must be the case, that if a man prefers the woman and that same woman prefers the same man, then these two would form a stable pair. It is obvious that the interest between the two parties are mutual and they should pair. Thus, in conclusion, the man must always be matched with his favorite woman preference.

Formally: Proving by contradiction, Suppose that there is a stable matching where M is ranked first on the preference list of all women and W is ranked first on the preference list of M with the output pairs (M, W') and (M', W) formed. Yet, since W is ranked first on the preference list of M, M must prefer W to W', and since M is the most popular man, W must prefer M to M'. Here we have an instability, as both M and W prefer each other to their current pairings, showing that this is a contradiction, thus, our claim holds.

2. Supposing that all women have the same exact preference ranking of men, using a similar argument from the previous part, the question establishes the fact that there is a "most popular" man, a "second most popular" man, a "third most popular" man, and so forth among all woman. If all women have identical rankings in the set, then it should be the case that there is a unique (optimal) stable matching, similar to that of the GS Algorithm. Basically, the problem suggests a unique instance of a set of Women that all arbitrarily chose the same set of preferences of man (very unlikely probability wise, but still the possibility was there).

   To formally demonstrate:

   Let us assume that there exists entity of men, **M** and **M'**, and entity of women **W** and **W'**.

   Attaching preferences, let the preference be that **M: (W, W'), M': (W, W'), W (M, M'), W'(M, M')** *Note (W, W') means that the Man prefers W to W'.

   **Here we can confirm that both woman share the same preference of a man.*

   Proceeding with the stable matching, to satisfy the stability condition, man M, must get his first choice W since it is so that the woman's first preference is M, creating the pair (M,W). As W is removed from the set of Woman to form possible pairs, it must be that M' pairs with the remaining woman W', forming the pair (M',W'). Thus at this point, we have a unique stable matching of (M,W) and (M',W') that are stable. *Note, it cannot be the case that both M and M' pick the same woman of first choice, since then the matching pairs will be unstable.

   Here, we can also further support the case where the preference of M' is (W', W) rather than the former. Here, it is still such that the pairs (M, W) and (W', M') will form, still producing a unique stable matching.

   We can extend this instance with more Men and Women as we seem fit, as we will describe in problem 3.

   More formally,

   We use a "building-up" proof, and prove this case by induction. For the base case, if we have 1 entity of Man and 1 entity of Woman, obviously there's only 1 stable matching so it's unique. We can say this for all possible sets of 1 Man and 1 Woman, that there's always a unique stable matching. Now if we look at 2 Men (M, M') and 2 Women (W, W'), we can match the most popular man to his best pick, and now we are left with 1 Man and 1 Woman that need to be matched. But based on what we concluded just now, all possible sets of 1 Man and 1 Woman always have a unique stable matching, so we know that they can be matched. And so forth, we now know that we have a unique stable matching for the overall set of 2 Men and 2 Woman since we have the smaller one from before, plus the one we just formed. Thus, all possible sets of 2 Men and 2 Women will have a unique stable matching. Now we can move on to 3 Men and Women and carry on this pattern. The most popular man will have his top pick, -> we're left with 2 Men and 2 Woman, and this is the same instance that we had before. Thus, what were are doing here, is that we are assuming the predicate that a set of Men and Woman, form a unique stable matching P(n). Starting from the base case of a singular set of Man and Woman. And we are showing that P(n+1) holds true, building from prior instances.

Therefore, we prove that there is a unique stable matching.

**Unique in the sense that the output of the same set is always the same. The solution/output may vary if the preferences of the sets varied.*

3. Given that all women have identical rankings, we can use this to our advantage and produce a simpler algorithm than Gale-Shapley, computing a unique stable matching.

   Such an algorithm involves that rather than continuing to loop for every free man that exists and loop in order of that man's preference of woman (taking O(n^2)) time, we loop through the set of men (in order of the preference as stated by Woman, since they share the same ranking). The algorithm follows that the most popular Man can choose whoever he wishes since all Women in the set desire him, and the second most popular man can choose whoever he wishes afterward since all Women prefer him to the others, and so forth. As we know that every woman shares identical rankings, it must be that the most popular man can choose whichever woman he prefers (his first choice). This pairing will be stable as stated in Prompt 1, since there is no pairing that could contradict this (they both share mutual interest). Then following, the second most popular man, can choose the woman highest on his preference list, since all woman prefer him to the other choices of male candidates. This process continues until all male candidates are exhausted to the least desirable man. Here we can describe the algorithm running at worst-case O(n) (a single loop), since we have the information of a static preferences of woman that shares no discrepancies.

   To prove that the matching is "stable", in the case that the current man's highest preferred woman is unavailable and taken by another (the previous) man, it must be the case that the Woman preferred the previous man to himself. This should be the case, as our algorithm describes, as we are looping from the most popular man to the least popular man, and this instance shows the unfortunate case that the most popular man shared the same highest woman preference as the man next in line. Therefore, that former pairing constructed was a valid stable pair as they both preferred each other to other options. Thus fulfilling the stability condition that no other pair could be preferred to the once outputted by the algorithm. The current man has no choice, but to pursue his second option and so forth. However, by the same logic, this Man still ranks highest on the woman's preference list given that the other man (the previous Man) has already found his pair. Therefore, this new pairing constructed will still be a stable pair (best matching).

   As a result of continuing this simpler algorithm, the output is a stable matching, building on the idea that previous matches must have been stable up to this point.

Question 2

In class (and in the textbook), we saw an example of a stable matching instance in which there are two different stable matchings. Building on this idea, give a class of instances such that for every k, you have an instance that has at least k distinct stable matchings. Explain clearly what these different stable matchings are, and why they are stable.

[Hints: (1) Obviously, as k gets large, your instances will need to contain more men/women. (2) The easiest way to solve this draws heavily on the example from the class/textbook. Think powers of two!]

Answer:

**As later seen, by "opposite preferences," I mean that if M has a preference list of (W1,W2...,W99) then the corresponding W has a preference list of (M99, M98,... M1).*

The instance in which there are two different stable matchings can be represented by the case where all entities of both Men and Woman prefer someone else, rather than each other. In this case of a stable matching, the choice is whether to make the women or man happy.

To demonstrate, consider an entity of men, M and M', and an entity of women, W and W'. Consider that the preferences of men and women are as follows:

M:(W, W'), M':(W', W), W:(M',M), W':(M, M').

Here we are struck with a dilemma. If we were to make the man happy, we would form the stable pairing (M, W), and (M', W'), completely disregarding the women's preference. If we were to make the woman happy, we would form the stable pairing (M', W), and (M, W'). We say stable because in both these cases, either the man is happy and would prefer no one else, or the woman is happy and likewise. Here we can see that there are two different stable matching outcomes, that both satisfy the "stable" condition.

Building on this idea, a class of instances such that for every k, we have an instance that has at **least** k distinct stable matchings, could be a case of an extension of the phenomenon as shown above; where each entity of man and woman have opposite preferences of each other.

As demonstrated above, if we are given two sets of entities of both Man and Women with completely opposite preferences, we have the situation that there can be two distinct stable matchings that either satisfies the man's preference or woman's preference. We also see later that there are more cases that we can consider. We know that this pairing is stable because at each case, both men are as happy as possible (therefore neither would leave their assigned partner), and likewise for the other case, both women are as possible (leading neither to leave their assignment as well).

Extending this phenomenon to k+1 instances (similar to an informal proof by induction), instead of going to 3 pairs, let us skip to 4 pairs. Here we "double" the 2 men and 2 woman example, meaning that we have the previous M, M', W, W' who have the opposite preferences from each other. Then we also have M3, M4, W3, and W4 who also have opposite preferences. We know that the subset of people (M, M', W, W") have two stable matchings as from the previous proof. And so does the subset of (M3, M4, W3, W4) (it is only that the entities names are only different, yet we still yield the same result). In total, there are 4 stable matchings. And we can assume from our conditions that W1 and W2 will probably rank lower in the preference list of M3 and M4, thus being irrelevant. We can further extend this principle, by lets say "tripling" the first example. Here since we have 6

Men and 6 Women, we can assume that there will be AT LEAST 6 stable matchings given k = 6. Here we are given a case that is being proved by Induction and building up from what we found in the previous case.

Suppose that we have an example of an odd case where there are 3 entities of Men (M, M', M") and 3 entities of Woman (W, W', W").

Let the preferences of each be that:

M: (W, W', W"), M': (W", W, W'), M": (W', W", W).

W: (M", M', M), W': (M', M, M"), W": (M, M", M')

**This set demonstrates the instance that each entity has a preference list completely opposite than each other.*

By following the rules and definition of a stable pair, we see that there is such an instance of three possible stable solutions to the given set.

If we were to satisfy male preference, we would get the stable matching set (M,W), (M', W"), (M", W'). We see that this pairing is stable because men are as happy as possible, so neither would leave their matched partner, thus satisfying the stable condition.

If we were to satisfy female preference, we would get the stable matching set (W, M"), (W', M'), (W", M). We also see that this pairing is stable because woman are as happy as possible, so neither would also leave their matched partner in this case... thus another pairing that satisfies the definition of stability.

Here, we can also consider another case where each Man and Woman get their second preference (the average choice). By following this condition, we would get the stable matching set (M, W'), (M', W), (M", W"). We see that this pairing is stable because both parties of men and female are content with their choices. It is stable because any other match that does not follow this rule, will lead to problems and inconsistencies of likeliness and preference... it is fine if everyone gets their second choice (everyone is equally content).

Here, we describe a general example of the class of instances in which for every k, we have at least k distinct stable matchings. The general description of the solution for every K is that there will always be a possible stable solution for making the Man most happy, there will always be a possible stable solution for making the Woman most happy, and there will always be stable solutions of ties that could prove to fulfill the stable condition as well. These ties being that any changes to these formed "tied" pairs, will prove to produce an instability, as there will be problems and inconsistencies of preferences among Man and Woman. Similar to what we did to prove the even case, we can attempt attempt to double this case similar to the other proof. Now, as we have M1,M2,M3... M6 and W1,W2,W3.. W6, we know that the subset of people M1-M3 and W1-W3 will form 3 unique stable matchings, and likewise, the subset of people M4-M6 and W4-W6 will also form 3 unique stable matchings (similar idea to even case). Summing, we get that we have 6 total unique stable matchings. Again, here we are given a case that is being proved by Induction and building up from what we found in the previous case. As K gets bigger, surely, the combination of such possible stable ties will grow as well.

As we showed a case where there were at least 2 distinct stable matchings with two entities of Man and Woman, and a case where there were at least 3 distinct stable matchings with three entities of Man and Woman, and a case where there were at least 4 distinct stable matchings with 4 Men and Women, we find here an instance where for every k, we can potentially get at least k unique stable matchings by following the set of preferences described and using the previous results to build off from. These conditions being that for every entity of Man and Woman, each entity has the complete opposite (reverse) set of preferences, than preferred (they have the complete opposite pairing from the opposite gender).

*As K gets large, the set of Men/Women also get larger. The set of possible combinations of stable pairings increase.

Question 3

We will look at another variant of "restricted" Stable Matching instances. Suppose that for each pair m,w, there is a happiness value H(m,w) if the two were paired up. Importantly, both m and w would experience the same happiness H(m,w) if they were matched. To avoid tie breaking issues, let's assume that all the H(m,w) values are different. Now, each man/woman will rank the other side by decreasing happiness they would get.

1. Show that for this type of input, there is again a simpler algorithm for finding a stable matching, which does not need to break any engagements. Describe such an algorithm, and prove that it finds a stable matching.

2. Suppose that our goal were to maximize the total happiness of all people, i.e., the sum of all happinesses of the matches that the matching contains. Give and explain an example where neither Gale-Shapley nor your algorithm from the previous part finds the happiness-maximizing matching.

Answer:

1. For this type of input, such an algorithm more simpler for finding a stable matching not needing to break any engagement, is an algorithm that loops through the happiness value of either Man and Woman, and pairs M and W accordingly to the partner that yields the highest happiness value. In this instance it matters that the set of Man or Woman iterated yields the highest happiness value to prevent any problems with inconsistencies. We can do this by sorting the entire set of happiness values and starting from the top, since it is so that both M and W would experience the same happiness value if they were matched. However, it is important that the choice of the first person matters. It is crucial that we pick the highest happiness value that exists so we do not run into the issue of breaking off ties and engagements. Therefore, it is important that the pairing with the highest happiness value go first in decreasing order. *Repetitive but crucial.

   ***The prompt states that every H(M, W) (Happiness Values) are different so we do not have to worry about ties. The prompt also states that each man/woman will rank the other side by the decreasing happiness they would get, implying that the happiness**

values are in sorted order going from the pair that forms the most happiness to the pair forming the least happiness.

Since the formal conditions are stated, we can think of Question 3 as similar to a preference list in the sense that there is a preference ranking among the entities of Man and Woman, except that now, we have a quantifiable happiness metric to go off of. Since there are no ties (every happiness values are different) and we have a universal metric, the situation given by Question 3 is a little easier to understand and implement.

To formally describe the algorithm and prove that it finds a stable matching:

Suppose that we have a set of Men (M, M', M",...) and a set of Woman (W, W', W",...). Between the set of Man and Woman, all the entities within the two sets have a happiness value corresponding their preference towards each member in the opposite set.

We choose some Man or Woman, and since each man/woman rank the other side by decreasing happiness, we simply choose the opposite gender entity that yields the highest happiness value and form a valid pair among the two (this value must be globally the highest value). We can achieve this by sorting the entire happiness values in order and starting from the top of the list. This pair is valid/stable because both the entity of Man and Woman experience the same happiness if they were matched, and with this algorithm, we explicitly choose the pairing that yields the highest happiness (this is important). Again, these two entities have no desire to break their engagement and will not prefer another potential suitor; the current pairing they have, satisfies their maximum potential happiness from the metric.

Since every happiness values among pairs are different, as stated in the prompt, we do not have to worry about a case of another man proposing to a woman that is already paired up and taken or a tie. This is because, since we are using the **HIGHEST** happiness value as a metric between Man and Woman, rather than analyzing the preference list, this pairing of man and woman would probably have yielded a lesser happiness value than this value, and thus, this man had no chance with the Woman. The Woman already paired up with whom she was most content with, and she is as happiest as she can be through the algorithms output, and likewise the other Man as well. Therefore, this algorithm finds a stable matching, because it never produces an instability where a Man and Woman that formed a pairing, desire to be matched with another valid pairing.

Example: Let there be a set of Man and Woman, attaching happiness values to each other. M(W(100), W'(80), W"(40)) | M'(W (90), W'(60), W"(100)) | M"(W(75), W'(95), W"(60)). W(M(100), M'(90), M"(75)) | W'(M(80), M'(60), M"(95)) | W"(M(40), M'(100), M"(60)).

**We can confirm that each valid pairing among M and W experience the same happiness value if they were matched.*

Here, the algorithm will simply pair (M,W) since they share the highest happiness value of 100 with each other. We also made sure that both M and W's highest happiness value are 100, and that neither entity has preference that yields a higher value. Then (M',W") since they share the highest happiness value of 100 with each other. And (M", W') since they share the highest happiness value of 95. All these pairs are stable matchings because each pairing yields the highest happiness level, and no entity would prefer a matching to another suitor.

Thus, algorithm proved.

2. Supposing that our goal was to maximize total happiness of all people, an example of where neither the GS algorithm nor the algorithm suggested outputs the correct solution is shown below.

   **Note, that the question asks just "give and explain example."*

   **We assume that all happiness values are different to avoid tie breaking issues.*

   GOAL: Maximize total happiness of all people.

   For both the GS Algorithm and the Algorithm shown above:

   Imagine a scenario of Men and Women such that we have a Set of Men (M, M', M",...) and a set of Woman (W, W', W",...). Between the set of Man and Woman, all the entities within the two sets have a happiness value corresponding their preference towards each member in the opposite set.

   Let the happiness value attached to set of Man and Woman be that:

   M(W(10), W'(9), W"(8)) | M'(W (7), W'(6), W"(2)) | M"(W(5), W'(4), W"(1)).

   W(M(10), M'(7), M"(5)) | W'(M(9), M'(6), M"(4)) | W"(M(8), M'(2), M"(1)).

   **We confirm there are no ties.*

   **We can assume that the GS Algorithm and the algorithm shown above is the same. Basically, the algorithm shown above is the GS algorithm in disguise, however, we use the happiness metric as the preferences instead. GS Algorithm formally describes a preference list, while the Q3 Algorithm states compatibility as a finite value.*

   By the algorithm, we should surely get the pairs (M,W) = 10, (M',W') = 6, and (M",W") = 1. Summing these happiness values yields a total happiness of **17**.

   However, there is actually a happiness maximizing matching that the algorithm fails to identify for the goal of this portion of the question. We can see that a possible pair (M,W") = 8, (M',W) = 7, and (M", W') = 4, yields a total happiness of **19**. We can see that this total happiness is higher than the one the algorithm outputs. Here the underlying idea is that Men M' and M" absolutely hate the idea of being paired with W", and have terrible happiness values attached to her. On the other hand, M is not so far off on his preference for W" and although he would prefer the other Women, he wouldn't mind pairing with W" for the sake of the happiness of his other "friends." Therefore, here we give and explain an example of a case where the algorithm from the previous part fails to find the maximum TOTAL happiness matching.

   *We make sure we follow the algorithm concisely.