Problem Set 1: Stable Matching

Handed out Friday, August 23. Due at the start of class Friday, August 30.

Homework Information: Some of the problems are probably too long to attempt the night before the due date, so plan accordingly. No late homework will be accepted. Feel free to work with others, but the work you hand in must be your own.

Problem 1.(10 points) Decide whether you think the following statement is true or false. If it is true, give a short explanation. If it is false, give a counterexample.

(a) True or false? In every instance of the Stable Matching Problem, there is a stable matching containing a pair $(h,\ s)$ such that h is ranked first on the preference list of s, and s is ranked first on the preference list of h.

(b) True or false? Consider an instance of the Stable Matching Problem in which there exists a hospital h and a student s such that h is ranked first on the preference list of s and s is ranked first on the preference list of h. Then in every stable matching M for this instance, the pair $(h,\ s)$ belongs to M.

Problem 2.(10 points) In the stable-matching problem, suppose that all the hospitals share the same preference ranking for the students. For example, suppose that they all list student 1 first, student 2 second, and so on. Further, suppose that the studnets obtain a copy of this shared rank ordering.

The students meet and decide the final pairing that they desire. By using their knowledge of the hospital's preference ordering, can the students force the Gale-Shapley algorithm to produce their desired pairing? (The students cannot change the algorithm, but they can coordinate their preference lists.)

If possible, present the student's solution and explain why it works. If not, give an counterexample in the form of a pairing that students desire, but that no matter how the students present their preferences, they cannot cause Gale-Shapley to produce this pairing.

Problem 3.(15 points) Consider the following simplified (and extremely unfair) algorithm for the stable matching problem. As in the standard problem, there are n hospitals, and n students, and each hospital and each student has an n-element preference list that rank orders all the members of the group. This algorithm *ignores* the preferences of the students and simply pairs each hospital with the first available student on its list.

```
for (i = 1 to n) {
  let (s[1],...,s[n]) be the students of h[i]'s preference list (from high to low)
  j = 1
  while (j <= n and h[i] is ummatched) {
      if (s[j] is not yet matched){
          match h[i] with s[j] (and both are now matched)
      }
      else j = j+1
  }
}
```

(Note that in this algorithm, once a student accept's a hospital's proposal, the student will never break it off.) We will explore the correctness of this algorithm by answering the following questions.

(a) Is this algorithm guaranteed to produce a perfect matching (that is, is every hospital paired with exactly one student and vice versa)? If so, give a proof, and if not, give a counterexample and explain your counterexample.

(b) If your answer to (a) was "no", skip the rest of this problem. Otherwise, is the matching produced by this algorithm guaranteed to be stable? If so, give a proof, and if not, present a counterexample and explain your counterexample.

(c) If your answer to (b) was "yes" skip this part. Otherwise, suppose that all the students in this system have exactly the same sets of preferences, and in particular, they rank the hospitals in (decreasing preference) order $< h_1, h_2, ..., h_n >$. (Each hospital's list contains all the students, but otherwise each hospital's preferences are arbitrary.) Under this restriction, is the matching produced by this algorithm guaranteed to be stable? As before, either give a proof or present a counterexample.

(Note: Throughout the semester, whenever you are asked to present a counterexample, you should strive to make your counterexample as short and clear as possible. In addition to giving the input for the counterexample, briefly explain what the algorithm does when run on this input and why it is wrong.)