CSE 521: Design and Analysis of Algorithms
Assignment #1
Jan 7, 2004
Due: Wednesay, January 14

Reading Assignment: Kleinberg and Tardos, Chapters 1 and 2.

Problems:

1. Consider a town with n men and n women seeking to get married to one another. Each man has a preference list that ranks all the women, and each woman has a preference list that ranks all the men.

   The set of all $2n$ people is divided into two categories: *good* people and *bad* people. Suppose that for some number k, $1 \leq k \leq n-1$, there are k good men and k good women; thus there are $n-k$ bad men and $n-k$ bad women.

   Everyone would rather marry any good person than any bad person. Formally, each preference list has the property that it ranks each good person of the opposite gender higher than each bad person of the opposite gender: its first k entries are the good people (of the opposite gender) in some order, and its next $n-k$ are the bad people (of the opposite gender) in some order.

   (a) Show that there **exists** a stable matching in which every good man is married to a good woman.

   (b) Show that in **every** stable matching, every good man is married to a good woman.

2. We can think about a different generalization of the stable matching problem, in which certain man-woman pairs are explicitly *forbidden*. In the case of employers and applicants, picture that certain applicants simply lack the necessary qualifications or degree; and so they cannot be employed at certain companies, however desirable they may seem. Concretely, we have a set M of n men, a set W of n women, and a set $F \subseteq M \times W$ of pairs who are simply *not allowed* to get married. Each man m ranks all the women w for which $(m, w) \notin F$, and each woman w' ranks all the men m' for which $(m', w') \notin F$.

   In this more general setting, we say that a matching S is *stable* if it does not exhibit any of the following types of instability.

   (i) There are two pairs (m, w) and (m', w') in S with the property that m prefers w' to w, and w' prefers m to m'. *(The usual kind of instability.)*

   (ii) There is a pair $(m, w) \in S$, and a man m', so that m' is not part of any pair in the matching, $(m', w) \notin F$, and w prefers m' to m. *(A single man is more desirable and not forbidden.)*

(ii′) There is a pair $(m, w) \in S$, and a woman w', so that w' is not part of any pair in the matching, $(m, w') \notin F$, and m prefers w' to w. *(A single woman is more desirable and not forbidden.)*

(iii) There is a man m and a woman w, neither of which is part of any pair in the matching, so that $(m, w) \notin F$. *(There are two single people with nothing preventing them from getting married to each other.)*

Note that under these more general definitions, a stable matching need not be a perfect matching.

Now we can ask: for every set of preference lists and every set of forbidden pairs, is there always a stable matching? Resolve this question by doing one of the following two things: (a) Giving an algorithm that, for any set of preference lists and forbidden pairs, produces a stable matching; or (b) Giving an example of a set of preference lists and forbidden pairs for which there is no stable matching.

3. There are many other settings in which we can ask questions related to some type of "stability" principle. Here's one, involving competition between two enterprises.

   Suppose we have two television networks; let's call them AOL-Time-Warner-CNN and Disney-ABC-ESPN, or $\mathcal{A}$ and $\mathcal{D}$ for short. There are n prime-time programming slots, and each network has n TV shows. Each network wants to devise a *schedule* — an assignment of each show to a distinct slot — so as to attract as much market share as possible.

   Here is the way we determine how well the two networks perform relative to each other, given their schedules. Each show has a fixed *Nielsen rating*, which is based on the number of people who watched it last year; we'll assume that no two shows have exactly the same rating. A network *wins* a given time slot if the show that it schedules for the time slot has a larger rating than the show the other network schedules for that time slot. The goal of each network is to *win* as many time slots as possible.

   Suppose in the opening week of the fall season, Network $\mathcal{A}$ reveals a schedule S and Network $\mathcal{D}$ reveals a schedule T. On the basis of this pair of schedules, each network wins certain time slots, according to the rule above. We'll say that the pair of schedules (S, T) is *stable* if neither network can unilaterally change its own schedule and win more time slots. That is, there is no schedule S' so that Network $\mathcal{A}$ wins more slots with the pair (S', T) than it did with the pair (S, T); and symmetrically, there is no schedule T' so that Network $\mathcal{D}$ wins more slots with the pair (S, T') than it did with the pair (S, T).

   The analogue of Gale and Shapley's question for this kind of stability is: For every set of TV shows and ratings, is there always a stable pair of schedules? Resolve this question by doing one of the following two things: (a) Giving an algorithm that, for any set of TV shows and associated ratings, produces a stable pair of schedules; or (b) Giving an example of a set of TV shows and associated ratings for which there is no stable pair of schedules.

4. Peripatetic Shipping Lines, Inc., is a shipping company that owns n ships, and provides service to n ports. Each of its ships has a *schedule* which says, for each day of the month, which of the ports it's currently visiting, or whether it's out at sea. (You can assume the "month" here has m days, for some $m > n$.) Each ship visits each port for exactly one day during the month. For safety reasons, PSL Inc. has the following strict requirement:

   (†) No two ships can be in the same port on the same day.

   The company wants to perform maintenance on all the ships this month, via the following scheme. They want to *truncate* each ship's schedule: for each ship S_i, there will be some day when it arrives in its scheduled port and simply remains there for the rest of the month (for maintenance). This means that S_i will not visit the remaining ports on its schedule (if any) that month, but this is okay. So the *truncation* of S_i's schedule will simply consist of its original schedule up to a certain specified day on which it is in a port P; the remainder of the truncated schedule simply has it remain in port P.

   Now the company's question to you is the following: Given the schedule for each ship, find a truncation of each so that condition (†) continues to hold: no two ships are ever in the same port on the same day.

   Show that such a set of truncations can always be found, and give an efficient algorithm to find them.

   Example: Suppose we have two ships and two ports, and the "month" has four days. Suppose the first ship's schedule is

   port P_1; at sea; port P_2; at sea

   and the second ship's schedule is

   at sea; port P_1; at sea; port P_2

   Then the (only) way to choose truncations would be to have the first ship remain in port P_2 starting on day 3, and have the second ship remain in port P_1 starting on day 2.

5. (*)

   For this problem, we will explore the issue of *truthfulness* in the stable matching problem, and specifically in the Gale-Shapley algorithm. The basic question is: Can a man or a woman end up better off by lying about his or her preferences? More concretely, we suppose each participant has a true preference order. Now consider a woman w. Suppose w prefers man m to m', but both m and m' are low on her list of preferences. Can it be the case that by switching the order of m and m' on her list of preferences (i.e., by falsely claiming that she prefers m' to m) and running the algorithm with this false preference list, w will end up with a man m'' that she truly prefers to both m and

m'? Similarly, can a man end up with a better partner by falsely switching the order of women on his preference list?

Resolve both questions by doing one of the following two things in each case:

(a) Giving a proof that, for any set of preference lists, switching the order of a pair on the list cannot improve a man's [woman's] partner in the Gale-Shapley algorithm; or

(b) Giving an example of a set of preference lists for which there is a switch that would improve the partner of a man [woman] who switched preferences.