CS 577: Introduction to Algorithms — Homework 1

Out: 09/12/06 — **Due: 09/26/06**

Ground Rules

- Grading. You will be graded on the correctness as well as clarity of your solutions. You are required to prove any claims that you make. In particular, when you are asked to design an algorithm, you must argue its correctness and running time.
- Collaboration. You are allowed to discuss questions with other people in the class. However, **you must solve and write your answers yourself without any help**. You must also give explicit citations to any sources besides the textbook and class notes, including discussions with classmates. Solutions taken from other sources, even if cited, will receive no credit unless there is significant "value added". In cases of doubt, you may be asked to explain your answer to the instructor and this will determine your grade.
- Lateness. Please see the class webpage for details on the lateness policy.
- Start working on your homework early. Plan your work in such a way that you have the opportunity to put some problems on the back burner for a while and revisit them later. Good luck!

Problems

1. **(Mix 'n Match.)** Recall the stable marriage problem discussed in class. Let P and Q be two different stable pairings of n boys with n girls. Consider the following way to build a new pairing R from P and Q. For each boy, P pairs him with some girl g_P and Q pairs him with some girl g_Q; to construct R we give him his favorite of the two girls g_P and g_Q. (g_P and g_Q might be the same girl, which is fine.) It is not even obvious that R is a pairing—perhaps some girl will get matched with two boys. Strangely, R is not only a pairing, it is stable!

   (a) Prove that R is a pairing and that it is stable.

   (b) What if we gave each boy his least favorite girl? Prove that this results in a stable pairing, or give a counter-example.

2. Problem 1.7 in the text (Pg. 26–27).

3. **(Online Dating.)** A dating website invites its users to post their profiles online, and matches men with women based on the compatibility between their profiles. The website provides a 100% satisfaction guaranteed service—if a paired couple are satisfied with their match, the website charges them a fee and provides them extra information about each other, otherwise the website compensates them for the failed effort. The company's profits are therefore proportional to the number of successful matches that they are able to find.

   The company has a software that matches various features of a couple's profiles and determines the likelihood of their match being compatible. Likelihoods are expressed as integers, with a larger number indicating a higher possibility of a match than a lower number.

   Here is the problem that the company faces: given n men and n women, as well as likelihoods for all the n^2 possible couples, come up with a pairing that maximizes the sum of the likelihoods of the matches. For example, for the likelihoods given in Figure 1, the best matching has total value 55. Having no algorithmicists on their staff, the company uses a straightforward greedy algorithm to solve this problem—match the most likely couple, remove them from the system and repeat. In the example below, this algorithm produces a solution with value 35. Although this approach does not produce the best solution possible, it gets fairly close – its value is more than half of the optimal value.

   (a) Prove that the greedy algorithm **always** produces a matching whose total value is more than half the total value of an optimal matching.

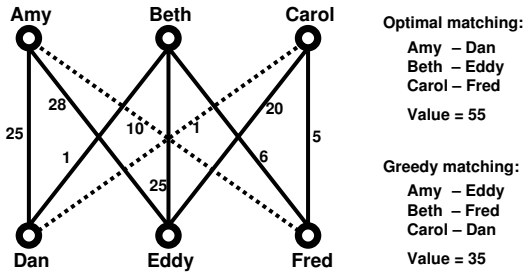

Figure 1: An online dating example

(b) Suppose that all the likelihoods are integers between the values 1 and W. One of the company's employees suggested the following variant to greedy matching in order to simplify the implementation of their matching software: pick an arbitrary ordering over women; match the first woman to the man with whom her likelihood is maximized; throw this pair out; match the second woman to the man with whom her likelihood is maximized among the remaining ones, and so on. In the above example, this algorithm gives the greedy matching when we use the ordering Amy, Beth and Carol, but gives the optimal solution when we use the ordering Beth, Amy and Carol. How bad can the performance of this algorithm be with respect to the optimal solution? Give the worst example that you can think of.

4. Problem 4.18 in the text (Pg. 197–198).

5. Problem 4.19 in the text (Pg. 198–199)

6. Problem 4.21 in the text (Pg. 200).