CSCI 270 Homework #4 Solutions

1. Write your name, student ID Number, and which lecture you attend (morning or afternoon). Multi-page submissions must be stapled.

2. You are driving across the country on a road trip, and are given a sorted list of integers $i_0 < i_1 < ... < i_{n-1}$ that correspond to the distance of each gas station from your starting location, and i_n being your destination. You can travel M miles on a full tank of gas. Write a greedy algorithm that minimizes the number of gas stations you stop at. Prove the correctness of your algorithm using a simplified exchange argument.
   Solution:
   Algorithm: Travel to the furthest station you can on each day.

   We'll prove that our first choice is correct, via contradiction. Assume that there is no optimal solution which travels to the furthest possible station on the first tank of gas.
   Consider some optimal solution OPT. OPT stops at some station i_o, where $i_{o+1} \leq M$. We stop at some station i_u, where $i_{u+1} > M$, so $u > o$.
   Tweak the OPT solution slightly (that is, make an exchange): keep OPT exactly the same except it stops at i_u first, instead of i_o, and call this solution OPT'.
   The first stop is valid, since $i_u < M$. The distance that OPT' travels until the second stop has been reduced by $i_u - i_o$, so the second stop is valid as well. All future stops remain unchanged. Thus OPT' is still a valid solution.
   In addition, the same number of stops have been used, so OPT' is also optimal. This contradicts our assumption, and proves our claim.

   After making the first choice, we can assume that we start at i_u instead of 0, and solve this reduced subproblem. That is, i_{u+1} becomes i_0, i_k becomes i_{k-u-1}, and all i values are reduced by i_u. We know our first choice on this reduced subproblem will be correct, so we can inductively prove that our entire algorithm is correct.

   More rigorously, we can formulate an inductive argument as follows.
   Base Case: when $n = 0$, our solution is optimal since we either make it to our destination or not, the same as OPT.
   Inductive Hypothesis: assume our solution is optimal when $0 \leq n \leq k$.
   Now consider $n = k + 1$. We know that our first choice is correct, so after the first stop we are at i_u.
   Now consider the reduced subproblem starting at i_u and going to i_n. We can create this subproblem by setting i_{u+1} to i_0, i_k to i_{k-u-1}, and reducing all i values by i_u.
   This new subproblem has $k+1-u < k$ stations, so by the inductive hypothesis, we optimally solve this subproblem, proving that we correctly solve any problem with $k + 1$ stations.
   By induction, our algorithm is optimal.

3. You are again driving across the country on a road trip. The gas stations are at distances $i_0 < i_1 < ... < i_{n-1}$, with i_n being your destination. You can travel M miles on a full tank

of gas. However, each gas station $i : 0 \le i \le n-1$ has a service cost s_i. Your goal is to now reach your destination while minimizing the sum of service costs you incur.

(a) Propose 2 reasonable greedy algorithms to solve this problem.
Solution:
We could try the same algorithm as part (a): always travel to the furthest possible station.
We could also try traveling to the smallest service cost station within driving distance each day.

(b) Prove that both of your algorithms presented in part (a) are **incorrect**.
Solution:
To disprove the algorithm from part (a), consider $i_0 = 1, i_1 = M, i_2 = M+1$, where $M > 1$. So our algorithm will travel to i_1 and then to our destination. Let $s_0 = 1, s_1 = 2$. So our algorithm finds a solution of size 2, but the optimal solution goes to i_0 then to the destination, a solution of size 1. Thus our algorithm is incorrect.

To disprove the second algorithm, consider $i_0 = 1, i_1 = M, i_2 = M+2$, where $M > 1$. Let $s_0 = 1, s_1 = 2$. Our algorithm will travel first to i_0, then to i_1, then to our destination, a solution of size 3. The optimal solution will travel to i_1 then the destination, a solution of size 2. Thus our algorithm is incorrect.

(c) Give a dynamic programming solution to this problem, and analyze the running time.
Solution:
For convenience, we'll define $i_{-1} = 0$. Our questions will be: what station do I stop at next?
We'll let $OPT(j) =$ the minimum service cost possible from i_j to i_n.
$OPT(j) = \min_k s_k + OPT(k)$, over all $k : j < k \le n, i_k - i_j \le M$.
$OPT(n) = 0$.

Now transform our recursive solution into an iterative one.
We'll fill the array in from largest j to smallest.
The answer is stored at $OPT(-1)$.
The size of the array is n, and the time per element is also n, so the total runtime is $\theta(n^2)$. Note that the greedy algorithm could run in linear time.

4. With political tempers running high, the dating site PolitiMatch has seen a boom in business amongst its Washington D.C. clientele. There are n male users who each specify their political ideology m_i on a scale from 0 to 10 (0 indicating extremely liberal, and 10 indicating extremely conservative). There are n female users, whose political ideology is f_i. Fractions are allowed, so a user could specify 3.14 as their ideology number. PolitiMatch attempts to minimize the difference of the ideology of those matched. More rigorously, if we match m_i, f_j, the value of the match v_k is $|m_i - f_j|$. Minimize the sum of the value of all pairings $\sum v_k$.

Write a greedy algorithm to solve this problem, and prove the correctness of your algorithm using an exchange argument.
Solution:

Greedily pair the smallest m_i value with the smallest f_i value.

Wlog, assume that the m and f values are sorted: $m_i \leq m_{i+1}$, and that $m_1 \leq f_1$ (the proof is mirrored when the opposite is true). Assume there is no optimal solution that pairs m_1 with f_1. Consider an arbitrary optimal solution which instead pairs m_1 with f_i and f_1 with m_j. Create OPT' which pairs m_1 with f_1 and m_j with f_i. OPT pays $|m_1 - f_i| + |m_j - f_1|$ and OPT' pays $|m_1 - f_1| + |m_j - f_i|$ for these two pairs.
Case 1: $m_j < f_i$. We want to show:

$$OPT \geq OPT'$$

$$f_i - m_1 + |m_j - f_1| \geq f_1 - m_1 + f_i - m_j$$
$$|m_j - f_1| \geq f_1 - m_j$$

Therefore this solution is at least as good, and pairs m_1 with f_1.
Case 2: $m_j \geq f_i$. We want to show:

$$OPT \geq OPT'$$

$$f_i - m_1 + m_j - f_1 \geq f_1 - m_1 + m_j - f_i$$
$$2f_i \geq 2f_1$$

Therefore, this solution is at least as good, and pairs m_1 with f_1. Thus we have a contradiction, and we know there is an optimal solution which makes the same first choice that we do.

We know it is correct to pair m_1 with f_1, so this reduces the problem to matching $m_2...m_n$ with $f_2...f_n$. We know our first choice on this reduced subproblem is also correct, so we can inductively argue our algorithm is optimal.

More rigorously:
Base Case: when $n = 1$, there is only one possible pairing, so obviously our algorithm is optimal.
Inductive Hypothesis: our algorithm is optimal for $n \leq k$.
Consider $n = k + 1$. We know it is correct to pair (m_1, f_1). This gives us the reduced subproblem on $m_2...m_{k+1}$ and $f_2...f_{k+1}$, which is a $n = k$ size subproblem. By the inductive hypothesis, we optimally solve this subproblem. This proves our algorithm correctly solves an $n = k + 1$ size subproblem.
By induction, our algorithm is correct.

5. Professor Slacker has a stack of final exams to be graded, and he hates grading! Therefore, he just assigned a random grade out of $10n$ to every student in the class. Slacker's laziness is coming back to haunt him however, as a student has requested to see their final exam! There are n problems, each worth 10 points. Slacker gave this student G total points, and now has to justify the grade to the student. Looking over the exam, Slacker determines that, if he had graded properly, he should have assigned g_i points to problem i, but $\sum_i g_i > G$. Your goal is to come up with a point distribution h_i for each problem such that $\sum_i h_i = G$, and

the inappopriateness of the grading is minimized. The inappropriateness of a single problem i is $(g_i - h_i)^2$, and the inappropriateness of the grading is $\sum_i (g_i - h_i)^2$. You are allowed to assign fractions such as 3.14 for grades.

Write a greedy algorithm to solve this problem, and prove the correctness of your algorithm using an exchange argument.

Solution:

Let $A = \frac{(\sum_i g_i) - G}{n}$, that is the average difference we need for each problem. Set the grade for problem i to be $h_i = g_i - A$. That is, $g_i - h_i$ should be the same for all problems.

Sidenote: if you determine you can't have negative scores, then you would want to raise all negative scores to 0, and redistribute the points equally to all other problems.

Define the similarity between two solutions to be the number of problems which have the same grade for both solutions. Let OPT be the optimal solution with the most similarity to our solution. Since OPT does not equal our solution, there must be some problem i where OPT's grade for o_i is less than h_i and some problem j where OPT's grade o_j is greater than h_j. Assume wlog that $h_i - o_i \leq o_j - h_j$ (the proof works equally well when the opposite is true).

Define OPT' to be identical to OPT, except $o'_i = h_i$ and $o'_j = o_j + o_i - h_i$. Note that the grades for OPT' still add up to G, and that there are now more identical grades. So, if we can prove that OPT' has at least as good a solution size to OPT, we'll have our contradiction and proven the optimality of our algorithm.

For the swapped pairs, OPT has cost $(g_i - o_i)^2 + (g_j - o_j)^2$. For the swapped pairs, OPT' has cost $(g_i - h_i)^2 + (g_j - [o_j + o_i - h_i])^2$. We want to show:

$$OPT \geq OPT'$$

$$(g_i - o_i)^2 + (g_j - o_j)^2 \geq (g_i - h_i)^2 + (g_j + h_i - o_j - o_i)^2$$

This is horribly complicated however, let's try to rewrite this in terms of what we're really trying to show. We defined A to be the average difference we need for each problem, which is $g_i - h_i$, for any i. So we rewrite what we need:

$$(A + h_i - o_i)^2 + (A + h_j - o_j)^2 \geq A^2 + (A + h_j + h_i - o_j - o_i)^2$$

To further simplify this, we were raising the score for i by $h_i - o_i = X$ and we needed to lower the score for j by $o_j - h_j = Y$, but we only lowered it part of the way. Note that X and Y are both positive. So we rewrite what we need:

$$(A + X)^2 + (A - Y)^2 \geq A^2 + (A + X - Y)^2$$

$$2A^2 + 2AX + X^2 - 2AY + Y^2 \geq 2A^2 + 2AX - 2AY + X^2 - 2XY + Y^2$$

$$0 \geq -2XY$$

$$0 \leq 2XY$$

Since X and Y are positive, the claim is true. Thus our algorithm must be optimal.

In fact, $2XY$ is strictly greater than 0. This means that we haven't just treaded water, we've actively improved the solution. This implies that OPT wasn't even optimal at all. This is still a contradiction, and gives us what we want, but it gives us the additional information that our solution is unique. There is no other optimal solution other than the one our algorithm produces.