CPSC 320 Proofs: Exam Edition Sample Solution

1 How to do 320 proofs

Lesson 1: Capture the key points

Practice question (Q2.1, 2017W1 midterm 1): You're given an SMP instance where two men have the same preference list. Prove that the Gale-Shapley algorithm (with men proposing) run on this instance will take **more than** n iterations.

- An incomplete proof: *Since two men share the same preference list, they will both propose to the same woman first. She can only accept one of them, so one of the men will have to propose again to someone else.*
- A not-fully-justified proof: *To complete in n iterations, we need each man to propose exactly once and have him be accepted. When two men have the same preference list, we need at least one more proposal, and hence at least $n + 1$ iterations.*

What are the necessary points for a complete proof?

SAMPLE SOLUTION:

Let's look first at why the two proofs above aren't entirely correct. The first one is clearly on the right track, in that it justifies why there has to be an additional proposal. But what's missing here is: why does this mean there have to be more than n proposals? (Helpful tip: it's a little hard to completely prove that some quantity – in this case, the number of iterations – is greater than n without even **mentioning** n.)

Now, for the second one: this proof states that when two men have the same preference list, we need at least one more proposal. But it doesn't say why this is the case.

Note that the two proofs together give us everything we need: the second proof gives us the little bit of background knowledge that was missing from the first proof – namely, that Gale-Shapley only completes in n iterations if every man proposes exactly once; and the first proof provides the justification of the need for an extra proposal that was missing from the second one. So here's how a correct solution could look:

Proof: To complete in n iterations, we need each man to propose exactly once and have him be accepted. Since two men share the same preference list, they will both propose to the same woman first. She can only accept one of them, so one of the men will have to propose again to someone else, and we will therefore have at least $n + 1$ iterations.

Lesson 2: Be precise

Practice question (Q4.3, 2016W1 midterm): Prove that an articulation point in a simple, unweighted, undirected graph is **never** diametric.

- An on-track but vague proof: *An articulation point can't be diametric, because articulation points are in the middle of a graph (because they connect two components), while diametric nodes are at the edges.*

How can we "translate" the above proof attempt into the language of a proof?

SAMPLE SOLUTION:

As before, let's start by discussing why this proof isn't correct. As hinted at in the description, the intuition here is good but it's not expressed precisely enough for a proof. What does it mean to be "in the middle" of a graph? (Does every graph have a middle? Given a graph, how do you know what the middle nodes are?) And similarly, it's not clear what it means to be "at the edges" of a graph. In fact, this is arguably a worse example of imprecision, because "edges" are already defined in the context of a graph: they're the things that connect the vertices. But that clearly isn't what the writer of this proof meant.

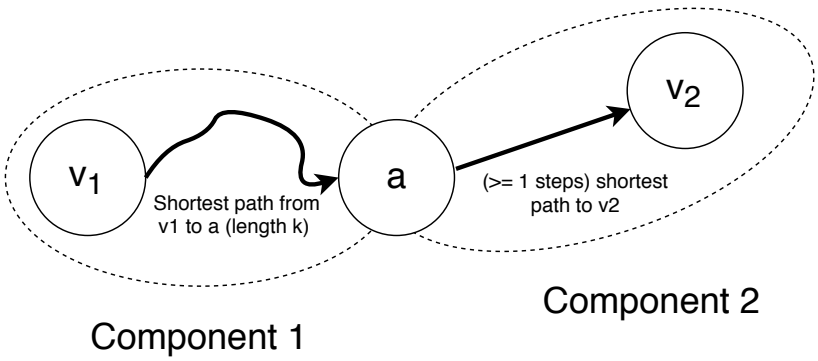
So let's try to refine the intuition behind this proof, given what we understand about articulation points and diametric nodes. We'll start with two observations:

- An articulation point connects (at least) two components of a graph, and removing it will separate these components. This means that a connects to at least one node in one component and at least one node in a second component, and that the only way to traverse from the first component to the second component is to go through a (if there were another way, removing a wouldn't separate the components, which would mean that a wasn't diametric).

- Diametric points are at the ends of a diametric path (i.e., a longest shortest path between any two nodes).

What I've done here is come up with more mathematical ways to express the ideas that "articulation points are in the middle" and "diametric points are at the edges." Now that we've come up with these, we'll write a proof by contradiction (there are other ways you could approach this question, but this one is the easiest).

Proof: Let a be an articulation point, and assume that a is a diametric node. Suppose the diametric path is from vertex v_1 to a, and it has length k. Because a is an articulation point, removing it would break the graph into two (or more) separate components. Without loss of generality, assume that v_1 is in Component 1. Component 2 has at least one node, which we'll call v_2. The shortest path from v_1 to v_2 must go through a, because otherwise removing a would **not** disconnect the two components. Because the shortest path from v_1 to a has length k and the path from v_1 to v_2 goes at least one step beyond a, we conclude that the path from v_1 to v_2 has length of at least $k + 1$. But this necessarily implies that a is not diametric, which completes the proof.



The figure above illustrates the main insight behind this proof. The trick is in realizing that any shortest path to a can be lengthened by extending it into the other component of the graph.

2 Exchange arguments (for greedy optimality proofs)

Exchange arguments are a technique for proving the optimality of greedy algorithms. Assume that we have an optimal solution to a problem that's different from the greedy solution. In an exchange argument, we show that we can gradually transform the optimal solution into our greedy solution without its quality ever decreasing. This shows that the optimal solution is no better than the greedy solution and the greedy solution is in fact optimal.

Steps in an exchange argument:

1. Define the greedy solution $\mathcal{G} = \{g_1, g_2, \ldots g_j\}$ and optimal solution $\mathcal{O} = \{o_1, o_2, \ldots o_k\}$.
2. Describe where $\mathcal{G}$ differs from $\mathcal{O}$. This could mean elements that appear in different orders in each solution, or elements that appear in one solution but not the other, etc.
3. Perform an exchange to transform the solution $\mathcal{O}$ into solution $\mathcal{G}$, proving that the quality will not decrease by doing so.

(Note: the swap described in steps 2 and 3 may need to be repeated multiple times, as in general there will be more than one swap that needs to be made. You need to be sure that you can exchange **repeatedly** without hurting the solution quality.)

Practice question (Q5, 2016W2 midterm)

I don't have the LaTeX source for the sample solutions for this question, and I don't want to write it all out myself. So I've appended the two pages of the 2016W2 midterm sample solution after this page. The solution includes the answers to all parts of this question (including the bonus), plus one additional part to this question that I didn't include in the worksheet.

5.1 Sample Solution

1. Give a small but non-trivial instance of the problem along with its optimal solution and the value (total fees) of that solution.

   5 dollars in the account. Charges of 3 dollars, 4 dollars, and 5 dollars. The optimal solution is 3, 4, and then 5 (although 4, 3, and then 5 results in the same answer), with a fee of $5^2 + 2^2 = 25 + 4 = 29$ cents.

2. Give pseudocode for a very simple greedy algorithm that solves the problem optimally in $O(n \lg n)$ time for n debits.

   Sort the debits in increasing order.

3. Complete the following proof of the correctness of your algorithm.

   We compare the greedy algorithm's debit ordering $\mathcal{G}$ against an optimal ordering $\mathcal{O}$. If $\mathcal{O}$ and $\mathcal{G}$ are the same, then $\mathcal{G}$ optimal. Otherwise, let d_i and d_{i+1} be a pair of neighboring debits that are in one order in $\mathcal{O}$ and the opposite order (and not necessarily neighbouring) in $\mathcal{G}$. Let the total of all the debits before d_i in O be T and the initial account balance be B. We now show that we can swap d_i and d_{i+1} without decreasing the overall value of the solution in four cases:

   Case 1, $B \leq T$ (i.e., both debits are entirely in overdraft): After swapping, both are still entirely in overdraft and so contribute the same amount to the fees collected.

   Case 2, $T < B < T + d_i$ (i.e., d_i is the debit that takes the account into overdraft) Since they're out of order with respect to the greedy solution, $d_i > d_{i+1}$ so $T + d_{i+1} < T + d_i$. There are then two interesting cases.

   (1) When $T < B < T + d_{i+1}$, debit d_{i+1} incurs some fees after the swap and d_i is entirely in overdraft after the swap. Before the swap, the total fees on these two debits are $d_{i+1}^2 + (T + d_i - B)^2 = d_{i+1}^2 + T^2 + d_i^2 + B^2 + 2Td_i - 2TB - 2d_iB$. After the swap, the fees are $d_i^2 + (T + d_{i+1} - B)^2 = d_i^2 + T^2 + d_{i+1}^2 + B^2 + 2Td_{i+1} - 2TB - 2d_{i+1}B$. Subtracting the pre-swap fees from the post-swap fees:

$$\begin{aligned}
\text{post-swap} - \text{pre-swap} &= d_i^2 + T^2 + d_{i+1}^2 + B^2 + 2Td_{i+1} - 2TB - 2d_{i+1}B - \\
&\qquad (d_{i+1}^2 + T^2 + d_i^2 + B^2 + 2Td_i - 2TB - 2d_iB) \\
&= 2Td_{i+1} - 2Td_i - 2d_{i+1}B + 2d_iB \\
&= 2T(d_{i+1} - d_i) - 2B(d_{i+1} - d_i) \\
&= 2(T - B)(d_{i+1} - d_i) \\
&= 2(B - T)(d_i - d_{i+1})
\end{aligned}$$

   We know $B - T > 0$ because $T < B$. We also know $d_i - d_{i+1} > 0$ because $d_i > d_{i+1}$. Thus, the post-swap fees are greater than the pre-swap fees. (Technically, that's a contradiction with this case being possible, since we're improving the optimal solution. However, for our purposes, the point is that the step doesn't reduce the value of the solution.)

   (2) When $T + d_{i+1} \leq B < T + d_i$, the second debit is entirely in overdraft before the swap and not at all in overdraft afterward, while the first debit is partly in overdraft before the swap and partly or entirely in overdraft afterward. For simplicity, we name the quantity $o = T + d_i + d_{i+1} - B$, which is the total amount of overdraft between the two debits.

   So, the pre-swap fees are $d_{i+1}^2 + (o - d_{i+1})^2 = d_{i+1}^2 + o^2 - 2od_{i+1} + d_{i+1}^2 = 2d_{i+1}^2 + o^2 - 2od_{i+1}$. The post-swap fees are just o^2, since the entire amount of the overdraft on these two debits is within d_i. Subtracting the pre-swap fees from the post-swap fees gives us: $2od_{i+1} - 2d_{i+1}^2 = 2d_{i+1}(o - d_{i+1})$,

which is greater than 0 since the total overdraft amount (o) was larger than d_{i+1}. Thus, again, this swap results in a solution that collects no less in fees than it used to.

(Note: clearly there is no change in the fees due to any other debit in the sequence when we make this swap.)

Case 3, $T + d_i \leq B < T + d_i + d_{i+1}$ **(i.e.,** d_{i+1} **is the debit that takes the account into overdraft)** Since $d_i > d_{i+1}$, when we swap these two debits, d_i will now be the debit that takes the account into overdraft. Since the total of the debits up to and including these two (now-swapped) debits remains the same, the amount of overdraft remains the same, and there is no change to the fees collected. (For some math to throw at this, note that $T + d_{i+1} < T + d_i \leq B$.)

(Note: clearly there is no change in the fees due to any other debit in the sequence when we make this swap.)

Case 4, $T + d_i + d_{i+1} \leq B$ **(i.e., neither debit is in overdraft)** When we swap the two debits, they're still both not in overdraft, and the fees collected remain the same.

Thus, we can start from $\mathcal{O}$ and repeatedly swap neighbouring debits that are in the opposite order to their order in $\mathcal{G}$ until the solution is identical to $\mathcal{G}$ without reducing the value (fees) of the solution, and so $\mathcal{G}$ is optimal. QED

4. Imagine the fee were $\frac{k}{100}$ instead instead of $\frac{k^2}{100}$. A friend proposes to you an algorithm that resequences the debits. Without knowing the details of the proposal, what can you say about how close that algorithm's result is to the optimal result? **Briefly and clearly justify your answer.**

   The total of the k values for all the debits is exactly the size of the negative balance. (Or, to put it another way, the total of all the debits is always the same, regardless of their order, and the total k values will be the total of the debits minus the balance, or zero of there are no overdrafts.) Thus, since all strategies produce the same negative balance, all strategies produce the same overdraft charge and are optimal.