# Practice Final Exam for Spring 2012

These problems are four of the seven problems from the final exam given in spring 2011, seven out of ten true or false questions, along with the full instructions given with the exam so that you have a sense of what the final exam will be like.

- Do not open this exam booklet until you are directed to do so. Read all the instructions first.
- The exam contains 7 problems, several with multiple parts. You have 180 minutes to earn 180 points.
- This exam booklet contains 10 pages, including this one, and two sheets of scratch paper which can be detached.
- This exam is closed book. You may use two double sided Letter ($8\frac{1}{2}'' \times 11''$) or A4 crib sheets. No calculators or programmable devices are permitted. Cell phones must be put away.
- Write your solutions in the space provided. If you run out of space, continue your answer on the back of the same sheet and make a notation on the front of the sheet.
- Do not waste time deriving facts that we have studied. It is sufficient to cite known results.
- Do not spend too much time on any one problem. Generally, a problem's point value is an indication of how many minutes to spend on it.
- Show your work, as partial credit will be given. You will be graded not only on the correctness of your answer, but also on the clarity with which you express it. Please be neat.
- Good luck!

| Problem | Title | Points | Parts | Grade | Initials |
|---|---|---|---|---|---|
| 0 | Name | 1 | 10 | | |
| 1 | True or False | 40 | 10 | | |
| 2 | Substitution Method | 14 | 1 | | |
| 3 | Updating a Flow Network | 25 | 3 | | |
| 4 | Candy Land | 20 | 1 | | |
| 5 | Combined-Signal Problem | 25 | 4 | | |
| 6 | Eleanor's Problem Revisited | 25 | 4 | | |
| 7 | Majority Rules | 30 | 5 | | |
| Total | | 180 | | | |

**Name:** _____

**Problem 0. Name.** [1 point] (10 parts)

Write your name on every page of this exam booklet! Don't forget the cover.

**Problem 1. True or False.** [40 points] (10 parts)

Circle **T** or **F** for each of the following statements to indicate whether the statement is true or false, respectively, and briefly explain why (one or two sentences). Your justification is worth more points than your true-or-false designation. *Careful:* Some problems are straightforward, but some are tricky!

**(a) T F** Suppose that every operation on a data structure runs in $O(1)$ amortized time. Then the running time for performing a sequence of $n$ operations on an initially empty data structure is $O(n)$ in the worst case.

**Solution:** True: $\sum_{i=1}^{n} \widehat{c}_i \geq \sum_{i=1}^{n} c_i$.

**(b) T F** Suppose that a Las Vegas algorithm has expected running time $\Theta(n)$ on inputs of size $n$. Then there may still be an input on which it always runs in time $\Omega(n \lg n)$.

**Solution:** False.

**(c) T F** If there is a randomized algorithm that solves a decision problem in time $t$ and outputs the correct answer with probability $0.5$, then there is a randomized algorithm for the problem that runs in time $\Theta(t)$ and outputs the correct answer with probability at least $0.99$.

**Solution:** False. Every decision problem has an algorithm that produces the correct answer with probability $0.5$ just by flipping a coin to determine the answer.

**(d) T F** Let $\mathcal{H} : \{0,1\}^n \to \{1,2,\ldots,k\}$ be a universal family of hash functions, and let $S \subseteq \{0,1\}^n$ be a set of $|S| = k$ elements. For $h$ chosen at random from $\mathcal{H}$, let $E$ be the event that for all $y \in \{1,2,\ldots,k\}$, the number of elements in $S$ hashed to $y$ is at most 100, that is, $|h^{-1}(y) \cap S| \leq 100$. Then we have $\Pr\{E\} \geq 3/4$.

**Solution:** False. $|h^{-1}(y) \cap S|$ is likely to be $\Theta(\log k / \log\log k)$ for some $y$. Only its expectation is $O(1)$.

**(e) T F** Let $\Sigma = \{a, b, c, \ldots, z\}$ be a 26-letter alphabet, and let $s \in \Sigma^n$ and $p \in \Sigma^m$ be strings of length $n$ and $m < n$ respectively. Then there is a $\Theta(n)$-time algorithm to check whether $p$ is a substring of $s$.

**Solution:** True. E.g., using suffix trees.

**(f) T F** If an iteration of the Ford-Fulkerson algorithm on a network places flow 1 through an edge $(u, v)$, then in every later iteration, the flow through $(u, v)$ is at least 1.

**Solution:** False: A later augmenting path may pass through $(v, u)$, causing the flow on $(u, v)$ to be decreased.

**(g) T F** There exists a minimization problem such that (i) assuming $P \neq NP$, there is no polynomial-time 1-approximation algorithm for the problem; and (ii) for any constant $\epsilon > 0$, there is a polynomial-time $(1 + \epsilon)$-approximation algorithm for the problem.

**Solution:** True. There are NP-hard optimization problems with a PTAS, such as PARTITION, as we saw in class.

**Problem 2. Substitution Method.** [14 points]

Use the substitution method to show that the recurrence

$$T(n) = \sqrt{n}\,T(\sqrt{n}) + n$$

has solution $T(n) = O(n \lg\lg n)$.

**Solution:** First, prove a base case. For $4 \leq n \leq 16$, let $T(n) = O(1) \leq k$ for some $k > 0$. For some $c \geq k/4$, we have that $T(n) \leq cn \lg\lg n$.
Now, prove the inductive case. Assume $T(n) \leq cn \lg\lg n$ for some $c > 0$. Then:

$$T(n) = \sqrt{n}\,T(\sqrt{n}) + n \leq \sqrt{n}\,c\sqrt{n}\,\lg\lg\sqrt{n} + n$$

We now find $c > 0$ so that

$$\begin{aligned}
\sqrt{n}\,c\sqrt{n}\,\lg\lg\sqrt{n} + n &\leq cn \lg\lg n \\
nc(\lg\lg n - \lg 2) + n &\leq cn \lg\lg n \\
-c \lg 2 + 1 &\leq 0 \\
1 &\leq c
\end{aligned}$$

Hence, the inductive case holds for any $c \geq 1$. Setting $c = \max\{k/4, 1\}$, we have $T(n) \leq cn \lg\lg n$ for all $4 \leq n$.

**Problem 3. Updating a Flow Network** [25 points] (3 parts)

Let $G = (V, E)$ be a flow network with source $s$ and sink $t$, and suppose that each edge $e \in E$ has capacity $c(e) = 1$. Assume also, for convenience, that $|E| = \Omega(V)$.

**(a)** Suppose that we implement the Ford-Fulkerson maximum-flow algorithm by using depth-first search to find augmenting paths in the residual graph. What is the worst-case running time of this algorithm on $G$?

**Solution:** Since the capacity out of the source is $|V| - 1$, a mincut has value at most $|V| - 1$. Thus the running time is $O(VE)$.

**(b)** Suppose that a maximum flow for $G$ has been computed using Ford-Fulkerson, and a new edge with unit capacity is added to $E$. Describe how the maximum flow can be efficiently updated. (*Note:* It is not the value of the flow that must be updated, but the flow itself.) Analyze your algorithm.

**Solution:** Simply run one more BFS to find one augmenting path in the residual flow network. This costs $O(E)$ time. One path suffices because all edges have capacity $1$ so any augmenting path will have capacity $1$ as well.
We could also precompute in $O(E)$ time for each vertex in the graph an edge on a path either to sink or from source in the residual flow network. (Can't have both if the flow is maximum!) Then, given the new edge we can update the flow in $O(V)$ time.

**(c)** Suppose that a maximum flow for $G$ has been computed using Ford-Fulkerson, but an edge is now removed from $E$. Describe how the maximum flow can be efficiently updated. Analyze your algorithm.

**Solution:** In the residual flow network, find a path from sink to source via the edge to be removed. To do so, find a path from sink to the starting endpoint of the directed edge in the residual flow network and similarly the second segment of the path. Then diminish the flow by one unit along the path. Since all edges have capacity $1$, this removes the flow from the edge to be removed. After removing the edge, search for augmenting path in the residual flow network. This is needed in case the edge was not in the minimum cut hence the flow can be redirected. The total cost is $O(E)$.

**Problem 4. Eleanor's Problem Revisited.** [25 points] (4 parts)

Recall the following problem abstracted from the take-home quiz, which we shall refer to as ***Eleanor's optimization problem***. Consider a directed graph $G = (V, E)$ in which every edge $e \in E$ has a length $\ell(e) \in \mathbb{Z}$ and a cost $c(e) \in \mathbb{Z}$. Given a source $s \in V$, a destination $t \in V$, and a cost $x$, find the shortest path in $G$ from $s$ to $t$ whose total cost is at most $x$.

We can reformulate Eleanor's optimization problem as a decision problem. ***Eleanor's decision problem*** has the same inputs as Eleanor's optimization problem, as well as a distance $d$. The problem is to determine if there exists a path in $G$ from $s$ to $t$ whose total cost is at most $x$ and whose length is at most $d$.

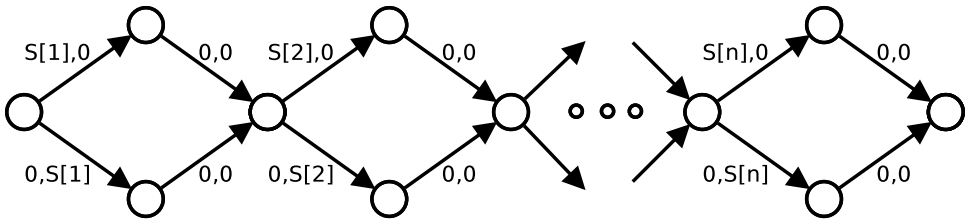
**(a)** Argue that Eleanor's decision problem has a polynomial-time algorithm if and only if Eleanor's optimization problem has a polynomial-time algorithm.

**Solution:** Reduce decision to optimization: solve optimization, output "YES" iff length of found path is less than $d$.
Reduce optimization to decision: Compute sum of all edge lengths which gives $d_{\max}$. If the decision algo returns "NO" for $d_{\max}$ then there is no solution. Now binary search for the smallest $d$ in the range $[0, d_{\max}]$ on which the decision algo returns "YES". Note $\log d_{\max}$ is polynomial in input size.

Recall the NP-complete PARTITION problem: Given an array $S[1\,..\,n]$ of natural numbers and a value $r \in \mathbb{N}$, determine whether there exists a set $A \subseteq \{1, 2, \ldots, n\}$ of indices such that

$$\max\left\{\sum_{i\in A} S[i], \sum_{i\notin A} S[i]\right\} \leq r\ .$$

**(b)** Prove that the Eleanor's decision problem is NP-hard. (*Hint:* Consider the graph illustrated below, where the label of each edge $e$ is "$c(e), \ell(e)$".)




**Solution:** Label the left-hand vertex in the hint graph as $s$ and the right-hand vertex $t$. Run the algo to the decision problem with this graph and $x = c$ and $d = c$ to finish the reduction. Any $s - t$ path in the graph corresponds to a set $A$ of the indices of those "widgets" where the path follows the upper branch. The length of this path is $\sum_{i\in A} S[i]$, while the cost of this path is $\sum_{i\notin A} S[i]$. The decision algo says YES iff there exists a path of both length and cost less than $c$, that is, the answer to the PARTITION problem is YES.

**(c)** Argue on the basis of part (b) that Eleanor's decision problem is NP-complete.

**Solution:** The witness is the path which can be checked in time linear in the number of its edges, which is less than $|V|$. Since the decision problem belongs to $\mathrm{NP}$ and is NP-hard, it is NP-complete.

**(d)** The solutions to the take-home quiz showed that there is an efficient $O(xE+xV \lg V)$ algorithm for Eleanor's optimization problem. Why doesn't this fact contradict the NP-completeness of Eleanor's decision problem?

**Solution:** An algorithm with runtime $O(xE + xV \lg V)$ has runtime polynomial in the **value** of $x$. In other words, the runtime is actually pseudo-polynomial, rather than truly polynomial. We showed that Eleanor's optimization problem was hard by a reduction from Partition, which is weakly NP-hard. As a result, we have only shown that Eleanor's Optimization Problem is weakly NP-hard, and so it's not a contradiction to have a pseudo-polynomial algorithm.

SCRATCH PAPER

SCRATCH PAPER

6.046J / 18.410J Design and Analysis of Algorithms
Spring 2012