MA/CSSE 473 – Design and Analysis of Algorithms

Homework 14 (84 points total) Updated for Summer, 2017

When a problem is given by number, it is from the textbook. 1.1.2 means "problem 2 from section 1.1" .

Problems for enlightenment/practice/review (not to turn in, but you should think about them):

How many of them you need to do serious work on depends on you and your background. I do not want to make everyone do one of them for the sake of the (possibly) few who need it. You can hopefully figure out which ones you need to do. **All problem numbers in this assignment (except #10) are the same in editions 3 and 4.**

9.4.6 (linear time Huffman code algorithm)
10.2.2a (maximum flow example)
10.2.5 (maximum flow algorithm for tree)
10.2.6a (prove equation 10.9)
10.4.3 (stable marriage example)
10.4.6 (unique stable marriage solution)
10.4.10 (roommate problem)

Problems to write up and turn in:

1. (5) 9.4.4 (maximal Huffman codeword length) Once you have figured out the answer, describe a set of probabilities (or frequencies) that make that maximum happen.

2. (10) 9.4.10 (card guessing)

3. (10) Answer the questions from class(below) about the induction proof of the correctness of Kruskal's algorithm. See details below.

4. (20) 8.3.11bc [8.3.10bc] (Matrix Chain Multiplication)

5. (5) 9.2.8 (efficiency of *find* in union-by-size)

   Problem 5 previous questions and answers from Piazza:

   Q: For quick union, with union by size, the following is used to describe the union (from the book):

   > The straightforward way for doing so is to always perform a union operation by attaching a smaller tree to the root of a larger one, with ties broken arbitrarily.

   How, specifically, does this deal with the subtrees of the larger tree's root?

   Also, are we supposed to consider path compression?

   A: To clarify, remember that the trees are not binary. They can have any number of subtrees.

   Why is there a need to deal with the larger tree's other subtrees? They are still there, and we are simply adding another one.

   For this problem, you do not need to deal with path compression.

6. (5) 11.1.1 (lower bound for alternating disk algorithm)

7. (5) 11.1.4 (fake coin minimum number of guesses)

8. (12) 11.1.10 (matrix multiplication and squaring) (6, 6)

9. (9) 11.2.10ab [11.2.8ab] (advanced fake-coin problem) (4, 5)

10. (5) 11.3.1 (Chess decidable?) Explain your answer.

11. (8) 11.3.2 (tractable?) Explain your answer.

Next page: Details on #3

At the end of this document: Additional Piazza Questions and answers from a previous term

Details of problem #3 Kruskal's algorithm

The questions (based on the information below)

(a) How do we know that v was already part of some connected component of G'?

Does the addition of e to C satisfy the hypothesis of the lemma? For each statement below, explain why it is true.

(b) G' is a subgraph of some MST for G:

(c) C is a connected component of G':

(d) e connects a vertex in C to an vertex in G – C:

(e) e satisfies the minimum-weight condition of the lemma:

The algorithm:

- To find a MST for a connected undirectedG:
  - Start with a graph G' containing all of the n vertices of G and no edges.
  - for i = 1 to n – 1:
    - Among all of G's edges that can be added without creating a cycle, add one (call it e) that has minimal weight.

The property we are trying to prove: Before every loop execution, G' is a subgraph of some MST of G.

Proof is (of course) by induction on i.
BASE CASE: When i = 1, G' consists of only vertices of G. Since all vertices must be part of every MST for G, G is a subgraph of every MST.

INDUCTION STEP. Assume that G' is a subgraph of an MST for G. Choose e according to the above algorithm. Show that $G' \cup \{e\}$ is a subgraph of an MST of G.

The Lemma we want to use: Let G be a weighted connected graph with a MST T; let G′ be any subgraph of T, and let C be any connected component of G′. If we add to C an edge *e*=(*v*,*w*) that has minimum-weight among all of the edges that have one vertex in C and the other vertex not in C, then G has an MST that contains the union of G′ and *e*.

In order to be able to use the lemma, we have to pick a connected component of C, and show that it satisfies the conditions of the lemma. We let v be one of the vertices of e, and let C be the connected component of G' that contains v.

Within this context, answer the 5 questions above.

Piazza Questions and answers from a previous term

#2 of HW14

You have a deck of cards that consists of one ace of spades, two deuces of spades, three threes, and on up to nine nines, making 45 cards in all. Someone draws a card from the shuffled deck.
1 + 2 + 3 + 9 = 15 does not equal 45.
I don't know where the rest cards come from in this question.

the instructors' answer: 1 + 2 + 3 + 4 + 5 + 6 + 7 + 8 + 9 = 45

#3 of HW14

Question all questions are simply inductive hypothesis or definition of G', C and e.

The essence of this proof is the lemma, but the questions seem to ask us to use this lemma without proving it, but the lemma itself already answers all of those questions.

I don't know what we are really asked to show.

Are we supposed to write a formal proof or just 'answer' them?

the instructors' answer:
What you are basically showing is that the situation of the theorem implies the hypothesis of the lemma, and thus implies the conclusion of the Lemma. The answers are mostly short and mostly obvious once you understand the definitions and the statement of the lemma. The purpose of this problem is mainly to allow you to check your understanding.