COMPSCI 320SC 2016 Test

Attempt *all three* questions. (Use of calculators is NOT permitted.)

Put the answers in the space below the questions. Write clearly and *show all your work*!

Marks for each question are shown below and just before each answer area.

This 50 minute test is worth 10% of your final grade for the course.

Question #:	1	2	3	Total
Possible marks:	10	10	10	30
Awarded marks:				

University ID: ______________________________

Student Name: ______________________________

Student Signature: ______________________________

Time Finished: ______________________________

1. (a) Who are the authors of the CompSci 320 textbook: *Algorithm Design*?

(***2 marks***)

Éva Tardos and Jon Kleinberg (okay if only family names known)

(b) Explain the *Stable Matching Problem* that was discussed in the first week of CompSci 320.

(***2 marks***)

Given preference profiles of n men and n women, find a stable matching. Here no man and woman prefer to be with each other than both their assigned partners.

(c) Name four algorithm design techniques (paradigms) that we will cover in CompSci 320?

(***2 marks***)

The main four are Greedy, Divide-and-Conquer, Dynamic Programming, Network Flow. Optionally allow Exhaustive search, Approximation, Randomization.

(d) Name three of the top (most influential) algorithms as mentioned in CompSci 320 lectures?

(***2 marks***)

Any three of: MergeSort, QuickSort, Fast Fourier Transform (FFT), Simplex method for LP, RSA cryptography, Dijkstra's SSP, Monte Carlo method.

(e) What are the names of the Lecturer and Tutor for CompSci 320?
Bonus (+1 mark): give the name of the Marker.

(***2 marks***)

Michael J. Dinneen and Zongcheng (Lucas) Yang, respectively. Bonus: Xiaojie Liu.

2. Consider the following divide-and-conquer "algorithm":

function printer(int n)
  for $i = 1$ **to** n^2 **do**
    $j = 1$
    while $j \leq \sqrt{i}$ **do**
      printline "hello world"
      $j = j + 1$
  if $n > 0$ **then**
    for $i = 1$ **to** 4 **do**
      printer($\lfloor n/2 \rfloor$)

Let $T(n)$ denote the number of lines of output generated by a call of printer(n). For the following, you may assume $\sum_{i=1}^{n} \sqrt{i} = \Theta(n^{\frac{3}{2}})$.

(a) What is the value of $T(1)$, $T(2)$, and $T(4)$? (*3 marks*)

$$T(1) = 1, \quad T(2) = 5 + 4 \cdot 1 = 9, \quad T(4) = 38 + 4 \cdot 9 = 74$$

(b) Provide a recurrence equation for $T(n)$. (*4 marks*)

$$T(n) = \begin{cases} 0 & \text{if } n = 0, \\ 4 \cdot T(\lfloor n/2 \rfloor) + c \cdot (n^2)^{\frac{3}{2}} & \text{if } n > 0, \text{ for some constant } c \end{cases}$$

(c) Solve the recurrence asymptotically for general n. (*3 marks*)

From the Master Recurrence Theorem, we have $a = 4$, $b = 2$, $k = 3$ so with $a < b^k$ we get $T(n) = \Theta(n^3)$.

3. We need to merge a set $S = \{f_1, f_2, \dots, f_n\}$ of sorted files of different lengths using an optimal *merging pattern* where the merging of two files f_i and f_j costs the sum of their lengths $|f_i| + |f_j|$. The total merging cost $C(T)$ of a merging pattern tree T is analogous to the Average Bits per Letter (ABL) cost of a prefix code ("Huffman Tree"). Here we sum the internal files/nodes (repeatedly merging) implicitly via the files' depths in the merging pattern:

$$C(T) = \sum_{k=1}^{n} |f_k| \cdot \text{depth}(f_k)$$

A greedy algorithm that finds an optimal merging pattern is as follows:

agorithm Merging_Pattern($S = \{f_1, f_2, \dots, f_n\}$)
    new PriorityQueue P
    for $i = 1$ **to** n **do**
        Store f_i in P indexed by its length $|f_i|$
    while P is not empty **do**
        (a) Extract two smallest elements f_i and f_j from P
        (b) Merge f_i and f_j and insert new file (parent node $f_{p=i+j}$) in P indexed by $|f_i| + |f_j|$

(a) What is the running time of this algorithm? Explain any implementation issues and indicate how to easily compute $C(T)$ from this algorithm. (***3 marks***)

$O(n \log n)$ using heaps to find the best merging pattern (repeat $O(\log n)$ min-heap operation $2n-1$ times). Note that $C(T)$ can be calculated by DFS to get depths of leafs.

(b) For files of lengths $|f_1| = 1, |f_2| = 2, |f_3| = 4, |f_4| = 4, |f_5| = 7, |f_6| = 9, |f_7| = 20, |f_8| = 25$ draw an optimal merging pattern as a rooted tree T and compute its cost $C(T)$. (***3 marks***)

Optimal cost is (1+2)*5+(4)*4+(4+7+9)+3*(20+25)*2= 15+16+60+90=181

(c) Show the correctness of this algorithm by giving a formal proof of this claim: For a set S there exists an optimal merging pattern such that the two shortest files f_i and f_j from S are first merged together. (*4 marks*)

We use an exchange argument. Suppose we have an optimal merging pattern T such that the two shortest files are not merged together. Consider the two deepest nodes f_a and f_b in this pattern. By assumption both f_a and f_b are not smaller than either of f_i and f_j. Create a new merging pattern T' by swapping f_a with f_i and f_b with f_j. Now $C(T') = \sum_{k=1}^{n} |f_k| \cdot \mathrm{depth}(f_k)$ is not greater than $C(T)$. We have constructed an optimal merging pattern that merges the two smallest files.

Scratch Page—will not be marked