CSOR W4246 – Fall, 2018

HW1 – Theoretical part (100 points)

Out: Friday, September 14, 2018

Due: 8pm, Tuesday, September 25, 2018

Please keep your answers clear and concise. You should always describe your algorithm clearly in English first and then give pseudocode.

You should write up the solutions **entirely on your own**. *Collaboration is limited to discussion of ideas only. Similarity between your solutions and solutions of your classmates or solutions posted online will result in receiving a 0 in this assignment and possibly further disciplinary actions.*

You must submit your assignment as a pdf file. Other file formats, such as jpg, doc, c, or py, **will not be graded, and will automatically receive a score of 0.** If you do not type your solutions, be sure that your hand-writing is legible, your scan is high-quality and your name is clearly written on your homework.

1. (20 points) Suppose we want to evaluate the polynomial $p(x) = a_0 + a_1x + a_2x^2 + \ldots + a_nx^n$ at point x.

   (a) Show that the following simple routine, known as *Horner's rule*, solves this problem and stores the solution in z.

   ```
   $z = a_n$
   for $i = n - 1$ down to 0 do
       $z = zx + a_i$
   end for
   ```

   (b) How many additions and multiplications does this routine use, as a function of n? Can you find a polynomial for which an alternative method is substantially better?

2. (20 points) The Hadamard matrices $H_0, H_1, H_2, \ldots$ are defined as follows:

   - H_0 is the 1×1 matrix $\begin{bmatrix} 1 \end{bmatrix}$
   - For $k > 0$, H_k is the $2^k \times 2^k$ matrix

   $$H_k = \begin{bmatrix} H_{k-1} & H_{k-1} \\ H_{k-1} & -H_{k-1} \end{bmatrix}$$

   Let ν be a column vector of length $n = 2^k$. Can you compute the matrix-vector product $H_k\nu$ faster than the straightforward algorithm that requires $O(n^2)$ time? Prove correctness and analyze the running time of your algorithm. (Assume that all the numbers involved are small enough that basic arithmetic operations like addition and multiplication take constant time.)

3. (25 points) Suppose that a sequence of items passes by one at a time. We want to maintain a sample of one item with the property that it is uniformly distributed over all the items that we have seen so far. Moreover we do not know the total number of items in advance and we cannot store more than one item at any time.

   (a) Consider the following algorithm. When the first item appears, we store it. When the k-th item appears, we replace the stored item with probability $1/k$. Show that this algorithm solves the problem.

   (b) Now suppose that when the k-th item appears, we replace the stored item with probability $1/2$. What is the distribution of the stored item in this case?

4. (35 points) You are given $n \times n$ matrices A, B, C and you wish to check whether $AB = C$.

   (a) (6 points) Give an $O(n^3)$ algorithm that solves this problem.

   (b) You will now design a faster randomized test (algorithm) for this task. Your test may fail to return the correct answer occasionally.

      i. (14 points) Let $\mathbf{x}$ be an n-dimensional vector with entries randomly and independently chosen to be 0 or 1, each with probability $1/2$. Prove that if M is a non-zero $n \times n$ matrix, then $\Pr[M\mathbf{x} = \mathbf{0}] \leq 1/2$.

      ii. (15 points) Show that $\Pr[AB\mathbf{x} = C\mathbf{x}] \leq 1/2$ if $AB \neq C$. Use this fact to design a randomized algorithm to check whether $AB = C$. (Your algorithm should be a Monte-Carlo algorithm, that is, it should always be efficient but may not always return the correct answer.)

      Analyze the time complexity of your algorithm and its success probability, that is, the probability that it returns the correct answer. How can you improve the success probability?

Recommended OPTIONAL exercises: do not return, they will NOT be graded!

1. In the table below, indicate the relationship between functions f and g for each pair (f, g) by writing **yes** or **no** in each box. For example, if $f = O(g)$ then write **yes** in the first box.

f	g	O	o	Ω	ω	Θ
$\log^5 n$	$10 \log^3 n$					
$n^2 \log(2n)$	$n \log n$					
$\sqrt{\log n}$	$\log \log n$					
$n^2 + n^{1/3}$	$n^2 \log n + n^{5/2}$					
$\sqrt{n} + 1500$	$n^{1/3} + \log n$					
$\frac{3^n}{n^2}$	$2^n \log n$					
$n^{\log n}$	2^n					
2^n	$\frac{3^n}{n^{\log n}}$					
n^n	$n!$					
$\log n^n$	$\log n!$					

2. Use the Master theorem to give tight asymptotic bounds for the following recurrences.

   - $T(n) = 4T(n/2) + n^2$.
   - $T(n) = 8T(n/2) + n^3$.
   - $T(n) = 11T(n/4) + n^2$.
   - $T(n) = 7T(n/3) + n$.

3. Consider the following recursive algorithm. On input a list of distinct numbers, the algorithm runs in three phases. In the first phase, the first $\lceil 2/3 \rceil$ elements of the list are sorted recursively; when the list has size 2, return the list if ordered, otherwise swap. In the second phase, the last $\lceil 2/3 \rceil$ elements are sorted recursively. Finally, in the third phase, the first $\lceil 2/3 \rceil$ elements are sorted recursively again. Derive a recurrence describing the running time of the algorithm and use the recurrence to bound its asymptotic running time. Would you use this algorithm in your next application?