Foundations of Computing

xx xx 2010

Fake exam! This is just a sketch!
Version 0.2

Thore Husfeldt, ITU

Instructions

What to bring. You can bring any written aid you want. This includes the course book and a dictionary. In fact, these two things are the only aids that make sense, so I recommend you bring them and only them. But if you want to bring other books, notes, print-out of code, old exams, or today's newspaper you can do so. (It won't help.)

You can't bring electronic aids (such as a laptop) or communication devices (such as a mobile phone). If you really want, you can bring an old-fashioned pocket calculator (not one that solves recurrence relations), but I can't see how that would be of any use to you.

1 Discrete maths only

1. Growth of functions.

(a) (*1 pt.*) What is the smallest integer n for which $f(x) = 2\log x + x^2$ is $O(x^n)$?

- A $n = 1$
- B $n = 2$
- C $n = 3$
- D $n = 4$

(b) (*1 pt.*) Which pair of functions as the property that f is $O(g)$ and g is $O(f)$?

- A $(x^3 + 3)^2$ and x^5
- B $x^2 + \log x$ and $(x + \log x)^2$
- C 2^x and x^2
- D $\dfrac{x}{x+1}$ and $\log x$

(c) (*1 pt.*) Which pair of values, C and k, are witnesses that show that $3x^2 + 26$ is $O(x^2)$?

- A $C = 3, k = 100$
- B $C = 5, k = 1$
- C $C = 2, k = 1000$
- D $C = 4, k = 6$

2. Let L be the language of strings over the vocabulary $\{0, 1\}$ that end in at least two 0s.

(a) (*1 pt.*) Which of the following deterministic automata reconises L?

- A (a drawing like on p. 808)
- B (another drawing)
- C (another drawing)

D (another drawing)

(b) (*1 pt.*) L is recognised by which regular expression?

A $(0 \cup 1)^*00$

B $(0 \cup 1)^*(0 \cup 00)$

C $*00$

D $(00 \cup 100 \cup 1^*00)$

... several more of this type of question, including about combinatorics, relations, languages, automata. The web-based self-assessments are a good training ground, at least for some of the chapters. That's where I've stolen the above examples as well.

3. The social network *FiveBook* allows you to have at most 5 *friends*. Alice can "propose" friendship to any number of members (except herself). As soon as another member (say, Bob) also proposes to her, Alice and Bob become "friends", except if either of them already has 5 friends. These concepts naturally define binary relations on the set of FiveBook members: the proposal relation P and the friendship relation F.

Answer this question on a separate piece of paper. Be brief but precise, you correct use of mathematical notation and formalism is an important aspect of this question.

(a) (*1 pt.*) Construct a very small social network, consisting of only 3 members, 3 proposals, and 1 friendship. Describe it both as in set-theoretic terms, as a graph, and as a matrix.

(b) (*1 pt.*) List the properties (reflexive, symmetric, transitive, etc.) that the relations P and F have and those that they don't have.

(c) (*1 pt.*) Is $P \subseteq F$? If no, give a counterexample.

(d) (*1 pt.*) Is $F = P \cap P^{-1}$? If no, give a counterexample.

(e) (*1 pt.*) Describe a situation where $(x, x) \in P^2 \setminus F$ for some x.

2 Algorithms and data structures only

4. Consider the following piece of code:

1: **for** $i = 1$ **to** n
2: **for** $j = 1$ **to** n
3: **print** $i + j$;

(a) (*2 pt.*) What is the running time? (Choose the smallest correct estimate.) Assume **print** takes constant time.

A $O(\sqrt{n} \log n)$

B $O(n)$

C $O(n^{3/2})$

D $O(n^2)$

(b) (*1 pt.*) Assume I changed line 4 to "**for** $j = 1$ **to** i". Then the running time for the whole algorithm is asymptotically

A faster

B slower

C the same

D question makes no sense

5. Assume you have a data structure that maintains a set S under insertion and deletion. The operation "**insert**(s, S)" makes sure that $s \in S$ holds, and the operation "**remove**(S)" chooses some $s \in S$ (assuming S is not empty) and removes it.

Consider the following piece of code, starting with an empty set S.

```
for i = 1 to n
    insert (i, S)
for i = 1 to n
    remove (S)
```

(a) (*1 pt.*) Assume **insert** (i, S) takes time $O(|S|)$, and **remove**(S) takes constant time. Then the running time is: (Choose the smallest correct estimate.)

A $O(n^{3/2})$

B $O(n^2)$

C $O(n^2 \log n)$

D $O(n^3)$

(b) (*1 pt.*) Assume I want the code to run in total time $O(n \log n)$. What bounds on the running time of insert and delete would guarantee this result?

A Both insert and delete must take time $O(\log |S|)$

B insert must take time $(\log |S|)$, as long as delete takes time $O(|S|)$

C delete must take time $(\log |S|)$, as long as insert takes time $O(|S|)$

D This is impossible, no matter what you do.

6. Consider the following piece of code:

```
int f(int n) {
    j = 0
    for i = 0 to n { j = j + i; }
    if j > 9 then { return f(n − 1) + n/2; }
    else return 3;
}
```

(a) (*3 pt.*) Which recurrence relation best characterises the running time of this method?

A $T(n) = T(n-1) + T(n/2)$

B $T(n) = T(n-1) + O(n)$

C $T(n) = 2T(n-1) + O(1)$

D $T(n) = 2T(n/2) + O(\log n)$

Excpect a number of exercises in the style of the "test your understanding" questions in Sedgewick (those are marked with an open triangle). For example:

7.

(a) (*1 pt.*) Consider a priority queue. A number i means "insert i" and "$*$" means "remove the maximum and output it:

$$3 * 35 * *41 * 3 * *$$

Give the sequence of values output by these operations.

A 3 5 3 4 3 1

B 3 3 5 4 1 3

C 3 1 4 5 3 3

D 1 3 3 3 4 5

(b) (*1 pt.*) Give the contents of the hash table that results when you insert items witht he keys E X A M Q U E S T I O N in that order inte an initually empty table of $M = 5$ lists, using separte chainign with unordered lists. Use the hash function $11k \bmod M$ to transform the kth letter of the alphabet into a table index.

A (a drawing in the style of fig. 14.6)

B another drawing

C another drawing

D another drawing

8. Consider the Super Vector Mario problem on page xxx.[1] This exercise asks you to suggest a solution to the problem — don't write a full programme! Write your answer on a separate piece of paper; use English or Danish; be brief and precise (do not use more than one page, preferably less). One concrete, complete, well-chosen example is often the best way of explaining yourself. If you want, you can use some form of code or pseudo-code, but you don't have to.

(a) (*5 pt.*) Explain how you would solve this problem in an efficient way. (Polynomial time will be fine.) Do you use a data structure or an algorithm? Which ones? What is the resulting running time and space usage?

[1] In the real exam, this of course will be a problem you have *not* seen before.