COMP 36111

Two hours

UNIVERSITY OF MANCHESTER
SCHOOL OF COMPUTER SCIENCE

Advanced Algorithms

Date: Monday 16th January 2017

Time: 09:45 - 11:45

Please answer any THREE Questions from the FOUR Questions provided

---

This is a CLOSED book examination

The use of electronic calculators is permitted provided they are not programmable and do not store text

[PTO]

1. Union find

The `union-find` algorithm takes as input a graph $G = (V,E)$ and returns a partition P of V giving the connected components of G.

```
begin union-find(V,E)
    let P = ∅
    for v ∈ V
       make(v)
    for (u,v) ∈ E
       if find(u) ≠ find(v)
          union(find(u),find(v))
    return P
```

Here, we assume that `make(`u`)` creates a cell $\{u\}$ and adds it to P, `find(`u`)` returns the unique element of P containing u, and `union(`C`,`D`)` replaces the cells C, D of P by a single cell $C \cup D$.

Suppose that the cells of the partition P are encoded as lists of vertices, and that vertices are encoded as data-structures with a `cell`-pointer giving the cell of the partition in which they lie. The operation `union` is given by

```
union(C,D)
    remove D from P
    for v ∈ D
       v →cell= C
    C= append(C,D).
```

a) Give the worst-case asymptotic time-complexity of this algorithm on a graph $G = (V,E)$ as a function of $|V|$ and $|E|$. (2 marks)

b) Suppose that this algorithm is run on the following connected graph, with the edges processed in order $e_1, \ldots, e_n$.

Note that e_i may be directed either as (v_{i-1}, v_i) or as (v_i, v_{i-1}). Give an asymptotic upper bound for the number of operations the algorithm will take in this case, if the best possible decisions are made as to the directions on the edges. Explain your answer. (2 marks)

c) Give an asymptotic lower bound for the number of operations the algorithm will take on the same graph, if the worst possible decisions are made as to the directions on the edges. Explain your answer. (2 marks)

d) Suppose now that the algorithm is modified so that `union` first checks the size of its arguments and, if necessary, interchanges them so that the second argument is no larger than the first. Show that the running time is bounded by $O(|E|+|V|\log(|V|))$. (6 marks)

e) Explain how, by representing cells of the partition as trees, the running time can be reduced to $O((|V|+|E|)\alpha(|V|))$, where $\alpha(x)$ is defined to be the smallest value y such that $A(y,y) \geq x$, and $A(m,n)$ is the so-called Ackermann function. You are not required to prove that your algorithm has the advertised complexity: it suffices to state clearly how `make`, `union` and `find` are implemented. You may use pseudocode and/or diagrams to explain these procedures. (8 marks)

2. Matching and flow optimization

a) Let $G = (U,V,E)$ be a bipartite graph. What is meant by a *perfect matching* for G? (2 marks)

b) Formally, a *flow network* is a quadruple $N = (V,E,s,t,c)$ where (V,E) is a directed graph, $s,t \in V$ and $c : E \to \mathbb{R}^+$. Explain the intuitive meaning of the components V, E, s, t and c. (2 marks)

c) State carefully the conditions required for a function $f : E \to \mathbb{R}$ to be a *flow* for $\mathbb{N}$. What is meant by the *value* of f? (4 marks)

d) Show how, given a flow network N and a non-maximal flow f, we can construct an auxiliary directed graph which will allow us to compute a larger flow. (6 marks)

e) Using pseudocode, give an algorithm for computing a flow of maximal value for a given network $N = (V,E,s,t,c)$ in the case where all the values of the function c are positive integers. You need not show the correctness of your algorithm. (2 marks)

f) Explain, using a diagram, how the existence of the algorithm in Question 2e can be used to show that the problem of determining whether a given bipartite graph has a perfect matching is in PTIME. (4 marks)

3. Linear programming and integer linear programming

   a) Define the problems *linear programming feasibility* (LP) and (non-negative) *integer linear programming feasibility* (ILP). (4 marks)

   b) In what well-known complexity classes are these problems known to lie? (2 marks)

   c) Consider a system of linear inequalities of the form

   $$A\mathbf{x} \geq \mathbf{b} \qquad\qquad \mathbf{x} \geq \mathbf{0}.$$

   Show that, if the numbers in A are all non-negative, this system is an instance of LP if and only if it is an instance of ILP. State any standard theorems that you use in your argument. (4 marks)

   d) Consider the system of linear equations and inequalities

   $$\begin{aligned} x_1 + x_2 + x_3 &= 3 & x_1 &\geq 0 \\ x_1 + x_2 &\geq x_3 & x_2 &\geq 0 \\ & & x_3 &\geq 0 \end{aligned}$$

   Sketch the bounded polytope of solutions, and determine its vertices. (6 marks)

   e) List the integer solutions of this system. (4 marks)

4. Non-deterministic logarithmic space

   a) Define the class of languages (problems) NLOGSPACE. (2 marks)

   b) The problem REACHABILITY is defined as

   Given: a directed graph $G = (V, E)$ and elements u_0, v_0 of V;
   Return: Y if v_0 is reachable from u_0 in G, N otherwise.

   Using pseudocode, give an algorithm showing that REACHABILITY is in NLOGSPACE. (4 marks)

   c) Explain how, by considering the configuration graph of a Turing machine, any problem in NLOGSPACE can be reduced, in logarithmic space, to an instance of REACHABILITY. (6 marks)

d) Recall that, in the context of propositional logic, a *literal* is a propositional variable or a negated propositional variable. A *Krom clause* is a formula $\ell \to m$, where ℓ and m are literals. The problem KromSAT is defined as

Given: a set Γ of Krom clauses;
Return: Y if Γ has a satisfying truth-value assignment, N otherwise.

Give a logarithmic space reduction of REACHABILITY to the *complement* of KromSAT, and conclude that KromSAT is Co-NLOGSPACE-hard. (6 marks)

e) Conclude that KromSAT is NLOGSPACE-hard. You may use any standard theorems about NLOGSPACE here, but you should state any such theorems carefully. (2 marks)

END OF EXAMINATION