

INSTITUTE OF COMPUTING - UNICAMP
Graduate Program
MO417A Design and Analysis of Algorithms
2015 - Semester 1 - Jorge Stolfi
Midterm exam 3 - 2015-06-29

Name

RA Number

Signature

Item													TOT
Points													

- You must do the exam by yourself, without any help.**
- You may not consult notes, books, tables, etc..**
- Computers and calculators may not be used during the exam.**
- Turn off and put away cellphones, music players, etc..**
- Do not separate the sheets of this exam booklet.**
- You cannot use any scratch paper besides this booklet.**
- Only answers in the marked-off spaces will be considered.**
- Purely numerical calculations need not be carried out.**
- Write your name legibly, and sign with a pen (not pencil or marker).**
- Once the exam booklet has been distributed:**
 - * if you leave the room, you will not be allowed to return.**
 - * after someone leaves the room, no one else will be admitted.**

1. A *semiautomaton with n states and d letters* is a directed graph G with vertices $V_G = \{0, \dots, n-1\}$, where each vertex has out-degree d , and the d outgoing edges of each vertex are labeled with distinct integers in the set $\{0, \dots, d-1\}$. Therefore, for each vertex v in V_G , and each integer k in $\{0, \dots, d-1\}$, we can denote by $\epsilon_G(v, k)$ the edge out of v with label k ; and by $\delta_G(v, k)$ the destination vertex of that edge.

Assume that the semiautomaton G is represented by a linked data structure, where each record rv represents a state (vertex) v of G . The record contains the fields

- $rv.ix$, the numeric index v of the vertex, in $\{0, \dots, n-1\}$;
- $rv.dst[]$, a list of d pointers, where $rv.dst[k]$ is a pointer to the record that represents the vertex $\delta_G(v, k)$, for k in $\{0, \dots, d-1\}$.

- (a) Suppose that a semiautomaton is represented by a set of linked state records, as above, except that the index field $.ix$ of every node contains arbitrary garbage. Give an algorithm *RenumberSA* that, given a pointer rs to one of these records, and the out-degree d , visits all state records that are reachable from that record, and stores in the $.ix$ field of each record a distinct sequential number, starting from 0. The algorithm should return the number n of vertices found, and a list $v = (v[0], v[1], \dots, v[n-1])$ of pointers to the records found, such that $v[k].ix = k$ for all k in $\{0, \dots, n-1\}$. Note that the algorithm does not know the number n in advance.

Algorithm *RenumberSA*(d, rs) returns (n, v)

<i>answer</i>

```

Rec(rs);
return (n, v).

```

Procedure *Rec*(rt)

<i>answer</i>

```

endproc;
endalg.

```

- (b) What is the running time of your algorithm, in the $O/\Theta/\Omega$ notation, as a function of n and d ?

2. Let G be a directed graph with vertices $\{0, \dots, n-1\}$, with no parallel edges. The graph G can be represented by its *adjacency matrix*, the $n \times n$ matrix of booleans G such that $G_{ij} = 1$ if and only if (i, j) is an edge of G .

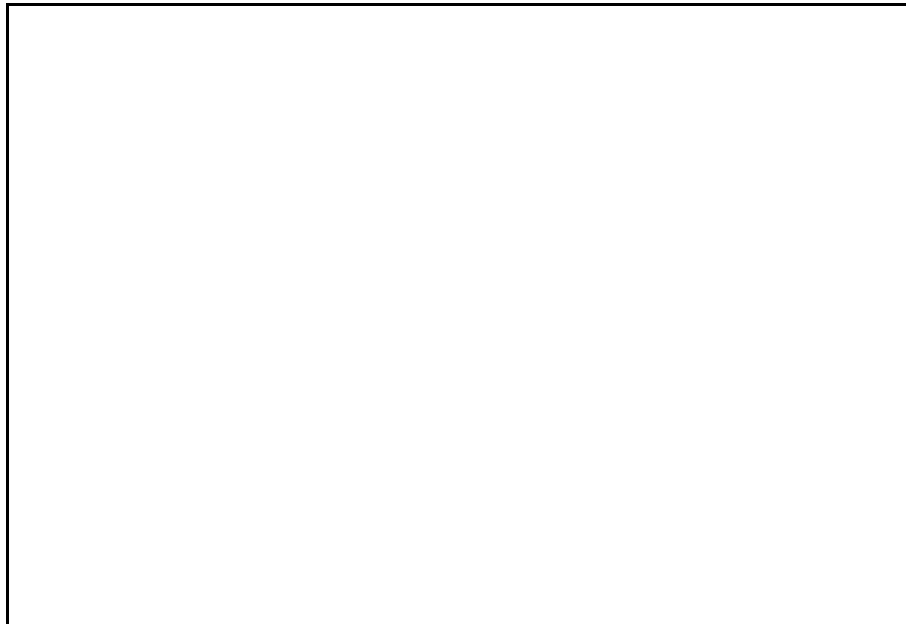
If A and B are boolean matrices with sizes $m \times n$ and $n \times p$, let $A \otimes B$ denote their boolean matrix product; that is, the $m \times p$ matrix C , such that

$$C_{ij} = \bigvee_{k=0}^{n-1} (A_{ik} \wedge B_{kj})$$

If A is an $n \times n$ boolean matrix, let $A^{(k)}$ denote the boolean product of A with itself k times; that is, $A^{(0)}$ is the $n \times n$ identity matrix, and $A^{(k)} = A \otimes A^{(k-1)}$ for any positive integer k .

A *walk in G with length m* is a non-empty sequence of vertices $P = (v_0, v_1, \dots, v_m)$ such that (v_{i-1}, v_i) is an edge of G , for all i in $\{1, \dots, m\}$. Note that the walk may go several times through the same vertex. The walk is *closed* if $v_0 = v_m$.

- (a) Write an algorithm $HasClosedWalk(n, G, m)$ that, given n , the graph G as an adjacency matrix, and a natural number m , determines whether G has a closed walk of length m . The algorithm must use boolean matrix multiplication. Try to get an algorithm that runs in time $O(mn^3)$.



- (b) A *circuit of length m in G* is a connected subgraph of G with m vertices and m edges, where every vertex has in-degree 1 and out-degree 1. It is not hard to prove that, if G has a closed walk with **odd** length m ($1, 3, 5, \dots$), then it has a circuit with some **odd** length p , where $p \leq m$. Write an algorithm $MinOddCirc(n, G)$ that, given n and a graph G as an adjacency matrix, determines the smallest **odd** natural number p such that G as a circuit with length p . The algorithm must use boolean matrix multiplication.

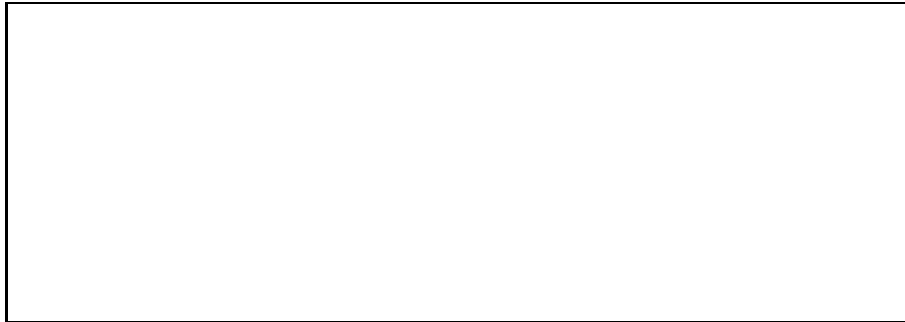
- (c) What is the running time of your algorithm for the item (b), in the $O/\Theta/\Omega$ notation, as a function of n ?

3. Suppose that n cities are connected by a network of telephone cables, such that there is at most one cable between each pair of cities. We can represent the network by an undirected connected graph G , with vertices $\{0, \dots, n - 1\}$, with an edge (i, j) for each cable between cities i and j .

Each cable between cities i and j may work all day with some probability P_{ij} , or break down during the day with probability $1 - P_{ij}$. In particular, $P_{ij} = 0$ if there is no cable between i and j .

Suppose that those cables cost money to keep working, so we would like to turn off as many of them as possible, while leaving all the cities still connected. That is, we want to find a spanning tree for G . Among all spanning trees, we want to find the one that has the highest probability of remaining connected all day.

- (a) What is the probability $Q(T)$ that a given spanning tree T of G will remain connected during the day?



- (b) Give an algorithm that finds the best T .

