

NAME: _____

ALGORITHMS QUALIFYING EXAM

This exam is open:

- books
- notes

and *closed*:

- neighbors
- calculators

The *upper bound* on exam time is 3 hours.

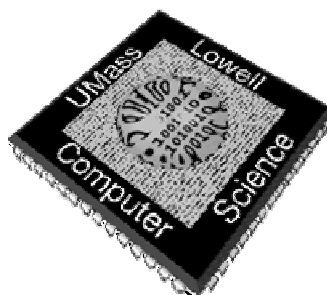
Please put all your work on the exam paper.

(Partial credit will only be given if your work is shown.)

If you use pseudocode from a reference, such as the Cormen *et al.* textbook, you can just give the page reference instead of writing that pseudocode on your exam paper.

All algorithms referred to here use a sequential model of computation, not a parallel one.

Good luck!



Part 1

(85 points)

Solve all of problems 1-7**1: (10 points) NP-Completeness**

Given: A directed graph $G = (V, E)$ and a positive integer $k \leq |V|$.

Question: Does there exist a subset $V' \subseteq V$ with $|V'| \leq k$ such that V' contains at least one vertex from each directed cycle in G ?

Prove that this problem is NP-complete.

this page left blank for work

2: (10 points) Flow Networks

Given a flow network $G = (V, E)$ with source s and sink t , a cut (S, T) is a partition of V into S and $T = V - S$ such that $s \in S$, $t \in T$. Let (S, T) be a cut of a flow network. Let A be a subset of S such that $s \in A$. Let $B = S - A$.

Consider the following statement about the flow network:

$$- f(T, S) = f(A, T) + f(B, V) + f(B, T)$$

where $f(X, Y) = \sum_{x \in X} \sum_{y \in Y} f(x, y)$ and $f(x, y)$ denotes the flow from vertex x to vertex y .

Either prove the statement or provide a counterexample.

this page left blank for work

3: (15 points) Maximum Spanning Tree

Design an efficient algorithm for finding a Maximum Spanning Tree of a weighted, connected graph. A Maximum Spanning Tree is a spanning tree with the largest possible total edge weight.

Provide pseudocode, correctness justification and upper and lower bounds on the worst-case asymptotic running time.

this page left blank for work

4: (15 points) Sequence Alignment

Consider two sequences of characters:

$$X = \langle x_1, x_2, \dots, x_m \rangle \quad \text{and} \quad Y = \langle y_1, y_2, \dots, y_n \rangle.$$

The sequence alignment problem here asks for the optimal alignment of characters of X with characters of Y , where optimality means maximal total cost. Gaps are allowed. The cost assumptions for an aligned pair of characters x_i and y_j are:

- if $x_i = y_j$, then this pair contributes cost c_1
- if $x_i \neq y_j$, then this pair contributes cost c_2
- if a character of one sequence is aligned with a gap in the other sequence, then this character contributes cost c_3

Example:

$X = \langle B, F, D, F, O, S, D, L, D, S, E, Q \rangle$

$Y = \langle B, W, D, O, F, S, D, L, M, D, L, Q, N \rangle$

For costs: $c_1 = 1$ $c_2 = 0$ $c_3 = 0$

The following alignment has total cost of 8, which is optimal:

```
BFD FOSDL DSEQ
BWDOF SDLMD LQN
```

For costs: $c_1 = 1$ $c_2 = -1$ $c_3 = -2$

The following alignment has total cost of 0, which is optimal:

```
BFDFOSDL DSEQ
BWDOFSDLMDLQN
```

Design an algorithm that solves the sequence alignment problem. Provide pseudocode, correctness justification and upper and lower bounds on the worst-case asymptotic running time.

this page left blank for work

5: (10 points) RSA Public Key Cryptosystem.

Consider an RSA key set with $p = 11$, $q = 31$, $n = 341$.

- (a) (2 points) Define $\phi(n)$ (the Euler symbol of n – you can refer to the correct page on the Cormen & al. textbook instead) and determine its value in this case.
- (b) (2 points) What is the smallest positive value of e (the public key) that can be used, and why?
- (c) (2 points) What is the value of d , the secret key, corresponding to this value of e ?
- (d) (2 points) What is the encrypted value of the message $M = 100$, using this smallest value of e ?
- (e) (2 points) What will the recipient of this message need to do to be able to read it?

Note: always indicate (no need to prove it) the theory behind your assertions: giving a set of numbers without explanation will result in a grade of 0.

this page left blank for work

6: (10 points) Amortized Analysis.

A sequence n of operations is performed on a data structure. The i -th operation costs i if i is an exact power of 3, and 1 otherwise.

Answer two of the three questions below:

- a) use aggregate analysis to determine the amortized cost per operation;**
- b) use the accounting method to determine the amortized cost per operation;**
- c) use the potential method to determine the amortized cost per operation.**

this page left blank for work

7: (15 points) Greedy Approximations.

Let (X, F) be an instance of the set covering problem (= a finite set X and a family F of subsets of X , such that every element of X belongs to at least one subset in F): $X = \bigcup_{S \in F} S$. We want to find a minimum size subset $C \subseteq F$ such that $X = \bigcup_{S \in C} S$. A greedy approximation algorithm is given by:

```

Greedy-Set-Cover( $X, F$ )
1  $U \leftarrow X$ 
2  $C \leftarrow \emptyset$ 
3 while  $U \neq \emptyset$ 
4   do select an  $S \in F$  that maximizes  $|S \cap U|$ 
5      $U \leftarrow U - S$ 
6      $C \leftarrow C \cup \{S\}$ 
7 return  $C$ 

```

One can prove that Greedy-Set-Cover is a polynomial-time $\rho(n)$ -approximation algorithm, where $\rho(n) = H(\max \{|S| : S \in F\})$, and $H(d)$ is the d -th harmonic number.

Give detailed proofs of the following statements, including descriptions of the data structures you need to support the operations:

- (2 pts) the number of iterations of the loop in lines 3-6 is bounded above by $\min(|X|, |F|)$;
- (2 pts) the loop body can be implemented to run in time $O(|X| \cdot |F|)$;
- (1 pt) Greedy-Set-Cover can be implemented so that it runs in time $O(|X| \cdot |F| \cdot \min(|X|, |F|))$;
- (5 pts) as a further improvement, Greedy-Set-Cover can be implemented so that it runs in time $O\left(\sum_{S \in F} |S|\right)$ Hint: you need the same data structures as before, plus a priority queue. Since X is finite, use a priority queue based on an array of size $|X|$ where each entry points to a (possibly empty) doubly linked list of sets whose cardinality is given by the array index (as well as being stored with the set itself). The "top" pointer is just the largest index with a non-null pointer. Show that this priority queue has

a cost of $O(1)$ for Insert, Remove-Max and Decrease-Key (the last two may only be amortized $O(1)$). Next, show that construction of the priority queue has cost $O\left(\sum_{S \in F} |S|\right)$ and then show that the cost of all loop iterations will not exceed a constant times $\sum_{S \in F} |S|$.

this page left blank for work

Part 2
(15 points)

Solve one of problems 8-10.

8: (15 points) Linear Programming

The set covering problem requires covering all the rows of an m -row, n -column 0-1-valued matrix A by a subset of the columns using minimum cost. A row is covered if at least one 1-valued element in the row is in a column that is used in the cover.

Let c_j be the cost of using column j in the cover. Assume that c_j is given for each $1 \leq j \leq n$ and that each c_j is positive.

Example:

$$A: \begin{bmatrix} 1 & 0 & 1 & 0 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \end{bmatrix} \quad \text{Costs: } (2, 3, 4, 5)$$

An optimal solution for this example selects columns 1 and 2, for a total cost of 5.

An integer program is a linear program in which one or more variables are constrained to have integer values.

Formulate an integer program that solves the set covering problem.

this page left blank for work

9: (15 points) Computational Geometry

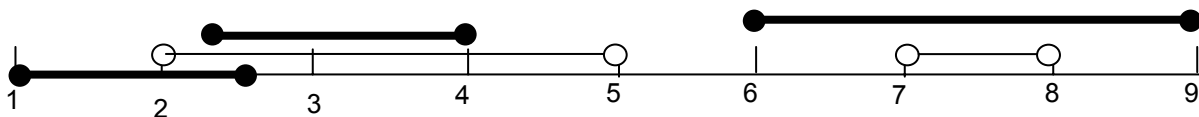
Given 2 collections R and B of 1D intervals on the x-axis of the form:

- $R = \{r_1, r_2, \dots, r_n\}$ where $r_i = (r_{si}, r_{fi})$ describes a single 1D interval for which r_{si} is the x-coordinate of the starting point and r_{fi} is the x-coordinate of the finishing point.
- $B = \{b_1, b_2, \dots, b_m\}$ is defined similarly in terms of $b_j = (b_{sj}, b_{fj})$.

The interval problem is to report the starting and finishing coordinates of each interval of the x-axis that is both in an interval of R and in an interval of B.

For example, for the figure below, the following results would be reported in the order given:

- An interval that is in both R and B starts at x=2.*
- An interval that is in both R and B ends at x=4.*
- An interval that is in both R and B starts at x=7.*
- An interval that is in both R and B ends at x=8.*



Prove the largest, worst-case asymptotic lower bound that you can on the running time required to solve this problem. Assume a sequential, algebraic decision tree model of computation.

this page left blank for work

10: (15 points) Matrix Multiplication, Transitive Closure, Induction

Let $M(n)$ be the time to multiply two $n \times n$ Boolean matrices, and let $T(n)$ be the time to find the transitive closure of an $n \times n$ Boolean matrix.

Show that an $M(n)$ -time Boolean matrix multiplication algorithm implies an $O(M(n) \lg n)$ -time transitive-closure algorithm, and a $T(n)$ -time transitive closure algorithm implies an $O(T(n))$ -time Boolean matrix multiplication algorithm.

You can present your own proof or you can follow the hints below.

Completing each of the first 3 hints will be worth 4 points.

Completing all of them will result in 15 points total, just as completing a valid independent proof.

Hints: a) Recall what is meant by transitive closure for a graph G , with vertices V and edges E . We consider a family of matrices $T^{(k)} = (t_{ij}^{(k)})$ constructed as follows:

```

Transitive - Closure( $G$ )
1.  $n \leftarrow |V[G]|$ 
2. for  $i \leftarrow 1$  to  $n$ 
3.   do for  $j \leftarrow i$  to  $n$ 
4.     do if  $i = 1$  or  $(i, j) \in E[G]$ 
5.       then  $t_{ij}^{(0)} \leftarrow 1$ 
6.       else  $t_{ij}^{(0)} \leftarrow 0$ 
7. for  $k \leftarrow 1$  to  $n$ 
8.   do for  $i \leftarrow 1$  to  $n$ 
9.     do for  $j \leftarrow 1$  to  $n$ 
10.      do  $t_{ij}^{(k)} \leftarrow t_{ij}^{(k-1)} \vee (t_{ik}^{(k-1)} \wedge t_{kj}^{(k-1)})$ 
11. return  $T^{(n)}$ 

```

The computation of $T^{(n)}$ is a $\Theta(n^3)$ -time operation, where $T^{(0)}$ is just the adjacency matrix of the graph G , and $T^{(n)}$ is the transitive closure.

A Boolean matrix A can be thus interpreted as the adjacency matrix for a graph. The identity matrix I_n has information about all the paths of length 0; A has information about all the paths of length 1. Prove by induction that, if the $n \times n$ Boolean matrix A is the adjacency matrix of a graph G , then $A^k \equiv A^{k-1}A$ (the k -th Boolean power of A) contains the

information about all paths of length k (for $1 \leq k \leq n-1$). The transitive closure of A is then $I_n + A + \dots + A^{n-1} = (I_n + A)^{n-1}$. Prove this latter result by induction on n . Recall that, for Boolean matrices, $A + A = A$.

b) Prove by induction that, if $P(n)$ is the fastest time to find the n -th power of an $n \times n$ matrix, then $P(n)$ is an $O(M(n) \lg n)$ -time algorithm. You may need to construct a recurrence relation.

c) What is the Transitive Closure of the Boolean matrix $\begin{bmatrix} I_n & A & 0 \\ 0 & I_n & B \\ 0 & 0 & I_n \end{bmatrix}$ where

A and B are Boolean matrices? Recall that, for Boolean matrices, $A + A = A$. Prove your claim.

d) Now connect the hints and complete the proof for the final 3 points.

this page left blank for work

this page left blank for work