CPSC 411: Design and Analysis of Algorithms
Exam 2
November 20, 2008

Name: Key

Instructions:

1. This is a closed book exam. Do not use any notes or books, other than your 8.5-by-11 inch review sheet. Do not confer with any other person. Do not use any computer equipment.
2. Show your work. Partial credit will be given. Grading will be based on correctness, clarity and neatness.
3. I suggest that you read the whole exam before beginning to work any problem. Budget your time wisely—according to the point distribution.
4. There are 4 questions worth a total of 100 points, on 7 pages (including this page).

DO NOT BEGIN THE EXAM UNTIL INSTRUCTED TO DO SO. GOOD LUCK!

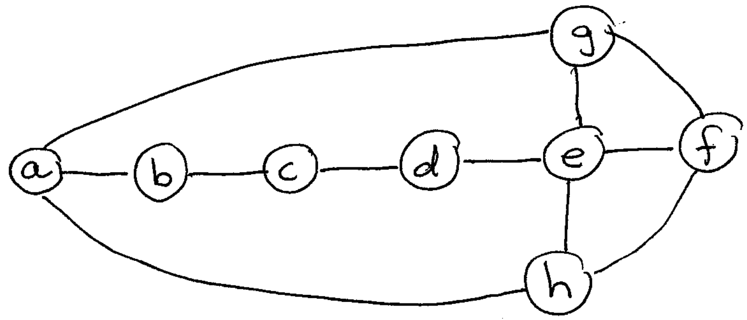
Please sign the academic integrity statement:

"On my honor, as an Aggie, I have neither given nor received unauthorized aid on this academic work. In particular, I certify that I have not received or given any assistance that is contrary to the letter or the spirit of the guidelines for this exam."

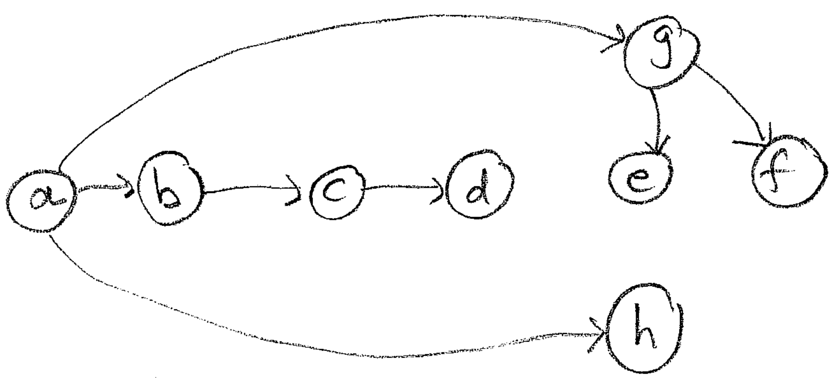
Signature: ______________________________

1. (30 pts total, 5 pts each) Graph Algorithms.

Consider the following undirected graph:



(a) Draw the breadth-first-search (BFS) tree resulting from running the BFS algorithm on the graph above, starting with node a. Assume that nodes are considered in alphabetical order. Use arrows to indicate the parent-child relationships.

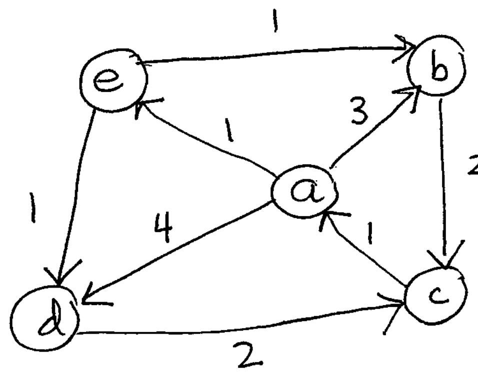


~~b~~ ~~g~~ ~~h~~ ~~c~~ ~~e~~ ~~f~~ ~~d~~

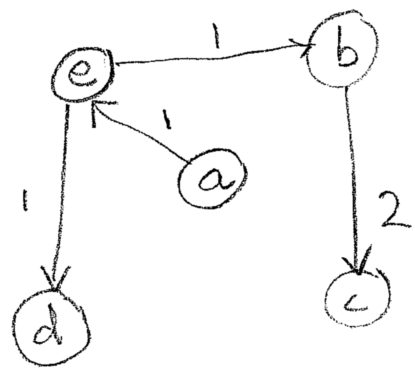
(b) What is the asymptotic running time of BFS on a graph $G = (V, E)$?

$O(V+E)$

Consider the following weighted directed graph:



(c) Draw a shortest path spanning tree for the graph above, using node a as the source.



(d) What is the asymptotic running time of Bellman-Ford's single source shortest path algorithm on a graph $G = (V, E)$?

$O(VE)$

(e) Assume that a binary heap is used for the priority queue in the implementation of Dijkstra's single source shortest path algorithm. What is the asymptotic running time of Dijkstra's algorithm on a graph $G = (V, E)$?

$O(V(T_{ins} + T_{ex}) + E \cdot T_{dec}) = O(E \log V)$

(f) Name one advantage of Dijkstra's algorithm over Bellman-Ford's algorithm, and one advantage of Bellman-Ford's algorithm over Dijkstra's algorithm.

Dijkstra's alg is faster

Bellman-Ford's can handle negative edge wts.

2. (25 pts total, 5 pts each) Randomized Algorithms.

(a) What is the definition of a random variable X?

function from sample space to real numbers

(b) What is the definition of the expectation, $E[X]$, of a random variable X?

$$E[X] = \sum_{\substack{\text{all values} \\ v \text{ that } X \text{ can take on}}} v \cdot Pr(X = v)$$

(c) What is the definition of an indicator random variable?

random variable that can only take on values 0 and 1 (indicating whether or not something happened)

(d) Prove that, if X is an indicator random variable, then $E[X] = \Pr[X = 1]$.

$$E[X] = \sum_{v} v \cdot Pr(X = v) = 0 \cdot Pr(X = 0) + 1 \cdot Pr(X = 1)$$
$$= Pr(X = 1)$$

(e) Consider a variation of the 3SAT problem, called MAX-3SAT, in which the goal is to find a truth assignment to the boolean variables that satisfies (causes to be true) as many clauses as possible. Assume that the three literals in each clause involve three distinct variables. Consider the following randomized algorithm for MAX-3SAT:

```
input:  3SAT formula with m clauses over n variables
1.  for each variable u_i, 1 ≤ i ≤ n, do
2.  set u_i to true with probability 1/2 and to false with probability 1/2
```

What is the expected number of clauses that will be satisfied with this algorithm?

Hint: Let X be the number of clauses that are satisfied; your goal is to find $E[X]$. Use indicator random variables, one for each clause. Part of the solution requires calculating the probability that a clause is satisfied.

Let X_j be indicator r.v. that is 1 if clause j is satisfied and 0 otherwise, $1 \le j \le m$

Note that $X = \sum_{j=1}^{m} X_j$.

$$E[X] = E\left[\sum_{j=1}^{m} X_j\right] = \sum_{j=1}^{m} E[X_j]$$

$$= \sum_{j=1}^{m} Pr(X_j = 1) = \sum_{j=1}^{m} \frac{7}{8}$$

since ~~Prob~~ clause j is true is $1 -$ prob. all 3 literals are false, which is $1 - \frac{1}{2} \cdot \frac{1}{2} \cdot \frac{1}{2}$

$$= m \cdot \frac{7}{8}$$

3. (25 pts total) NP-Completeness.

(a) (4 pts) *True or False:* Suppose we know that there is a polynomial reduction from problem P1 to problem P2. If there is no polynomial-time algorithm for P2, then there is no polynomial-time algorithm for P1. Explain your answer.

$P1 \le_p P2$

no poly time alg? $\Leftarrow$ no poly time alg.

FALSE.

P2 is at least as hard and possibly harder than P1. So we don't know how hard P1 is.

Consider the decision problem BDST, for "bounded degree spanning tree": Given an undirected graph G and integer K, does G have a spanning tree in which no node has degree more than K? (The degree of a node is the number of neighbors.) Follow the outline given below to prove that BDST is NP-complete. For the reduction, assume that you already know that the Hamiltonian Path (HP) decision problem is NP-complete. The HP problem is this: Given an undirected graph G, is there a path in G that contains every node exactly once? (It is similar to the Hamiltonian cycle problem, but asks for a path instead of a cycle.)

(b) (5 pts) Show BDST is in NP.

Given a candidate solution to BDST (a set S of edges in G), check in poly. time if

1) S forms a spanning tree of G, and
2) each node has at most K neighbors using the edges in S.

(c) **(5 pts)** Which way should the reduction go: given any BDST input, construct a HP input, OR, given any HP input, construct a BDST input?

want to show $HP \leq_p BDST$

(d) **(5 pts)** Let f be the reduction. What relationship must hold between x and $f(x)$ regarding having or not having a bounded degree spanning tree and having or not having a Hamiltonian path?

x has a Hamiltonian path iff $f(x)$ has a bounded degree spanning tree. In particular, let $x = G(V, E)$ & $f(x) = [G'(V', E'), K]$ — then G has HP iff G' has K-bdd degree S.T.

(e) **(3 pts)** How fast must f be?

polynomial time

(f) **(3 pts)** Describe f and prove that it has the required properties to show BDST is NP-complete. *Hint: In your reduction, use a specific, small, value for K.*

Given arbitrary graph G as input to HP, create BDST input $(G, 2)$.

Claim: G has a H.P. iff G has a S.T. w/ degree at most 2.

Note that a spanning tree w/ degree at most 2 is exactly the same thing as a Hamiltonian path (a chain that includes all nodes).

4. (20 pts total) Approximation Algorithms.

(a) (7 pts) Show that if $C \subseteq V$ is a vertex cover of an undirected graph $G = (V, E)$, then $V - C$ is an independent set of G.

Suppose in contradiction $V-C$ is not an independent set of G. Then there are two nodes in $V-C$, x and y, with an edge between them. But then this edge (x,y) is not covered by C, since neither x nor y is in C (they are both in $V-C$). Contradiction.

Recall the approximation algorithm A_{VC} for the vertex cover minimization problem that has ratio bound 2 (i.e., the size of the vertex cover returned by the algorithm is at most twice as big as the size of the smallest possible vertex cover).

We will use A_{VC} to get an approximation algorithm for the independent set maximization problem using a reduction inspired by part (a). The independent set approximation algorithm, A_{IS}, works like this:

input: $G = (V, E)$
1. run A_{VC} on G, returning set C
2. return $V - C$ for the independent set of G

(b) (8 pts) Suppose G is a graph whose minimum vertex cover has size $\frac{|V|}{2} - 1$ and A_{VC} performs at its worst on G. What is the ratio bound achieved by A_{IS} on input G?

Since A_{VC} performs its worst on G, it returns a vertex cover of size $2\left(\frac{|V|}{2}-1\right) = |V| - 2$.

Then A_{IS} returns an independent set of size 2.

But the max. indep. set has size $\frac{|V|}{2} + 1$ (V − the min. vertex cover) + ratio bound is $\left(\frac{|V|}{2}+1\right)/2 = \frac{|V|}{4} + 2$.

(c) (5 pts) What does part (b) imply about the ability of A_{IS} to achieve a constant ratio bound approximation? Explain your answer.

Can't do it: ratio bound depends on size of input ($|V|$), not a constant.