Introduction to Computer Algorithms CS470
Final Examination
Monday, December 12, 2005
8:00am – 10:30am

Student name ……………………………………………………

Student number ………………………

There are five problems totaling 120 points. For full credit it suffices to obtain 90 points. Any points above 90 are extra credit. You can use three letter-size pages of your own notes, written on any of the two sides of the pages; no other material is permitted (e.g., no books).

For every algorithm that you design, precede the algorithm with an outline in English describing how the algorithm works (just a pseudocode is not sufficient).

Problem 1: [15 points]

Find an asymptotically tight bound on T(n) defined by the following recurrence

$$T(n) = \begin{cases} 1, & n = 1, \\ 1, & n = 2, \\ T(\lfloor n/2 \rfloor) + T(\lfloor n/3 \rfloor) + 4n, & n \geq 3. \end{cases}$$

Solution:

Problem 2: [15 points]

Consider the following algorithm operating on an array A[1…n] with a number b

```
Alg(n,A,b)
► n≥1 and A[1…n]
j ← 1
k ← 0
while j≤n
    if A[j]>b then
        k ← k+1
    j ← j+1
output k
```

1) State a natural computational problem that this algorithm solves in the form of allowed input n, A, b and a required output.
2) Use the technique of loop invariant to demonstrate that the algorithm is correct (i.e., that is solves the computational problem).

Solution:

Problem 3: [30 points]

Suppose that there are two friends Alice and Bob and a collection of n items denoted 1,…,n, where n is even. Alice and Bob would like to partition the items in half so as to maximize value in a certain precise sense. Specifically, the value that Alice assigns to item i is $v_A[i]$ and the value that Bob assigns is $v_B[i]$. The values are integers (possibly negative). Suppose that items $A \subseteq \{1,\ldots,n\}$ are assigned to Alice, and so items $\{1,\ldots,n\} \setminus A$ are assigned to Bob. The value *value(A)* of the partition A is defined as

$$value(A) = \sum_{i \in A} v_A[i] + \sum_{i \in \{1,\ldots,n\} \setminus A} v_B[i]$$

The goal is to find a set A of cardinality n/2 that maximizes *value(A)*, given $v_A[1],\ldots,v_A[n]$ and $v_B[1],\ldots,v_B[n]$.

For example, if the values of items are

$v_A[1] = 1$	$v_A[2] = -2$	$v_A[3] = 3$	$v_A[4] = 0$
$v_B[1] = 3$	$v_B[2] = 2$	$v_B[3] = 1$	$v_B[4] = 3$

then the answer should be A={1,3}.

Design an algorithm and present an asymptotic upper bound on the worst-case running time of the algorithm. Argue why your algorithm and upper bound are correct.

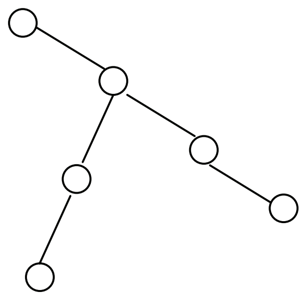
Hint: given a set A, when is exchanging an item from A for an item not in A advantageous?

Solution:

Problem 4: [30 points]

The *diameter* of an undirected graph G=(V,E) is the length of a longest simple path in the graph (i.e., where nodes are distinct along the path).

Example: given a graph



the answer should be 4.

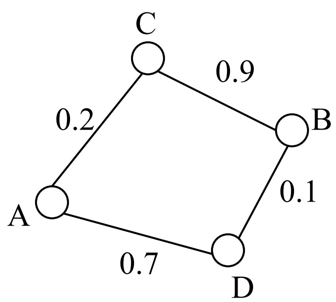
Design an algorithm that computes the diameter of any tree and present an asymptotic upper bound on the worst-case running time of the algorithm. Argue why your algorithm and upper bound are correct.

Solution:

Problem 5: [30 points]

Consider a collection of cities, where there is a direct road between some pairs of cities. Roads may be in bad shape. If you travel along the road from u to v, then you succeed with probability $0<p_{u-v}<1$ (your car does not break down), while with probability $1-p_{u-v}$ you fail. The successes are independent for different roads.

For example, given the following graph that models the problem



If we travel from A to B through C, then the probability of success is 0.2·0.9 = 0.18, while if we travel through D instead, then the probability is 0.7·0.1 = 0.07. Hence it is better to travel from A to B through C, rather than through D.

In general, we are given an undirected graph G=(V,E) where each edge u-v has a *success probability* p_{u-v}, $0<p_{u-v}<1$, and two distinct nodes A and B. Given a path $P=v_0,\ldots,v_k$, the probability of success of reaching v_k from v_0 along the path P is

$$succ(P) = \prod_{i=0}^{k-1} p_{v_i - v_{i+1}}$$

We want to find a path Q from A to B that maximizes the probability of success of reaching B from A.

Design an algorithm that computes a path Q that maximizes the probability of success of reaching B from A, given the graph and probabilities. Present an asymptotic upper bound on the worst-case running time of your algorithm. Argue why your algorithm and upper bound are correct.

Solution: