CMPT 307-d1 FINAL
April 18, 2006

PART A

1. (10 points) Indicate for each pair of expressions (A, B) in the list below, whether A is O, o, Ω or Θ of B. Assume that $k \geq 1$, $\epsilon > 0$, and $c > 1$ are constants. Your answer should be in the form of "yes" or "no".

$$(\log^k n, n^\epsilon), (n^k, c^n), (2^n, 2^{\frac{n}{2}}), (\log_2 n!, log_2(n^n)).$$

2. (15 points) What is minimum number of comparisons one has to make to sort 5 elements? Give an algorithm to sort 5 elements using the minimum number of comparisons.

3. (20 points)

A. The following is the usual algorithm for multiplying two $N \times N$ matrices.

```
procedure matix-multiplication(int A,B: out C:array[1..N,1..N])
   begin
      for I:=1 to N do
        for J:=1 to N do
        begin
           C[i,j]:= 0;
           for K:=1 to N do
              C[I,J] := C[I,J] + C[I,K]*C[K,J]
        end
   end.
```

What is the running time of the procedure?

B. There is an algorithm for multiplying two $N \times N$ matrices whose worst case running time satisfies the recurrence equaltion $T(N) = 7T(\frac{N}{2}) + \Theta(N^2)$. Solve the equation.

C. Which is better, the algorithm in part A or the algorithm in part B?

4. (20 points) Let us say that an array A of length N is *almost sorted* with errors of size k for $k < n$ if for any i, j, if $j - i > k$ then $A[j] \geq A[i]$. Thus the array does not have to be completely ordered but any two elements in the array that are out of order cannot be more than k places apart. For example the list

5 8 1 6 15 12 11 20 19 25 30 35 32

is almost sorted to error of size 2. A[3]=1 is less than A[1] = 5 and 3-1=2, but there are no elements out of order that are 3 or more steps apart.

2

(a) Show how quicksort can be modified to produce a list that is almost sorted with errors of size k. Assuming k is much less than N, how should the pivot be chosen? What is the best case running time of this modified quiksort?

(b) If the input array is almost sorted with errors of size k, what is the running time of the insertion sort? Justify your answer.

5. (10 points) Let $X[1..N]$ and $Y[1..N]$ be two arrays each containing n numbers already in sorted order. Give an $O(\log N)$-time algorithm to find the median of all $2N$ elements in arrays X and Y.

6. (15 points) You are given two 2-3-4 trees T and T', storing m and n keys respectively. Every key stored in T is smaller than evry key stored in T'. How can you merge the two trees into a 2-3-4 tree storing $m + n$ keys using only $O(\log m + \log n)$ time?

7. (15 points) Describe a data structure for a set S of integers of arbitrary size that supports the operations listed below with the given running times. Give a brief (no more than two lines) description of what is involved in each operation.

Operations	Comments	Running times
$INSERT(x, S)$ $DELETE(x, S)$		Worst case time $O(\log \vert S \vert)$
$Member(x, S)$		~~Average time $O(1)$,~~ Worst case time $O(\log \vert S \vert)$
$MIN(S), MAX(S)$ $DELETE(x, S)$		Worst case time $O(\log \vert S \vert)$
$COUNT(x, y, S)$	Returns the number of elements in S between x and y (inclusive).	Worst case time $O(\log \vert S \vert)$.
$NEXT(x, S)$	Given an element x of S, find the next larger element in S.	~~Average time $O(1)$~~ ~~and~~ worst case time $O(\log \vert S)$.

③

CMPT 307 (Evening) Final Exam
April 10, 2003

There are a total of 1[illegible]5 points. Answer any [illegible]5 points. You must answer questions 1 and 4. Please clearly identify the two questions that you don't want them to be marked.

1. [20 points]

   Specify **True** or **False** for each of the following statements to indicate whether the statement is true or false, respectively. If the statement is correct, then briefly state why. If the statement is wrong, explain why. Your justification has more points than just true-or-false designation.

   (a) Any n-node tree that satisfies both the min-heap property and the binary-search-tree property on the same n distinct keys has height $\Theta(n)$.

   (b) Number of distinct elements in an array of size **n** can be determined in $O(n)$ worst case time.

   ~~(c) Number of distinct elements in an array of size **n** can be determined in $O(n)$ expected time.~~

   ~~(d) For hashing an item into a hash table in which collisions are resolved by chaining, the worst-case time is proportional to the load factor of the table.~~

   (e) For any DAG, there is only one topological ordering of the vertices.

   (f) A heap with n elements can be converted into a binary search trees in $O(n)$ time.

   (g) The ~~breadth~~ depth first search algorithm makes use of a stack.

   (h) Whether an undirected graph $G = (V, E)$ contains a cycle can be detected in $O(|V|)$ time. (Assume that the adjacency list of the graph is already stored in the memory.)

2. [10 points]

   The asymptotic performance of a particular divide-and-conquer program can be described by the recurrence $T(n) = 5T(\frac{n}{2}) + \Theta(n^2)$. Because the structure of the program and its asymptotic performance cannot be changed, the programmer is constrained to improve the constant factor in its running time. Where should the programmer prioritize recording effort, and why? (For example, improving the code for the base of the recursion, the divide step, the combine step etc.)

4

3. [10 points]

   You are given two height balanced binary search trees T and T', storing m and n elements respectively. Every element of tree T is smaller than every element of tree T'. Every node u also stores height of the subtree rooted at it. Using this extra information how can you merge the two trees in time $O(logm + logn)$ (preserving both the height balance and the order)?

4. [35 points] **Short Problems**

   (a) Let P be a problem instance of size n. Let A be an algorithm to solve P. The worst case running time of the algorithm is $O(n^2)$. What are the worst case lower bound and upper bound of P ? What can you say about the optimality of A?

   (b) Consider a modification to QUICKSORT such that each time PARTITION is called, the median of the partitioned array is found (using the SELECT algorithm) used as a pivot. What is the worst case running time of QUICKSORT?

   ~~(c) Show that n integers each of value less than n^{100} can be sorted in linear time.~~

   (d) We are given an array with n entries that are supposed to be links to parents in a forest of parentpointer trees. Write an algorithm that checks in O(n) worst-case time whether these pointers indeed define a true forest of parentpointer trees.

   (e) Suppose that in an edge-weighted, undirected graph $G(V, E)$, the edge (u, v) is the unique edge incident to u having the least weight, that is, every other edge incident to u has strictly greater weight. Show that (u, v) belongs to every minimum spanning tree of G.

   (f) Given a set of n elements, show that one can output in sorted order the k elements following the median in sorted order in time $O(n + k \log k)$.

   (g) Show that the longest path in a dag $G = (V, E)$ can be found in $O(|V|+|E|)$ time.

~~5. [10 points]~~

~~Recall the "linked list" representation of disjoint sets from class.~~

~~Argue that the total time spent on performing n MAKE-SET and m UNION operations is $O(n + m \log n)$.~~

6. [10 points]

   Given a graph $G = (V, E)$, test whether the graph is bipartite. Your algorithm should run in optimal time. (A graph G is bipartite if V can be partitioned into V_1 and V_2 such that for any edge $e = (u, v)$ of G, either $u \in V_1$ and $v \in V_2$ or $v \in V_1$ and $u \in V_2$.)

7. [10 points]

   In general, a graph can have several minimum spanning trees. However, if the graph has distinct edge costs, one can show that it will have a unique minimum spanning tree. Prove the fact.

8. [10 points]

   Describe an efficient algorithm that, given an undirected graph G, determines a spanning tree of G whose largest edge weight is minimum over all spanning trees of G.

CMPT 307 Final (3 hours)
May 22, 2004

Questions 1 and 2 must be answered. From the remaining, answer questions worth 70 points.

1. (10) Answer any one of the following:

   (a) The Fibonacci numbers F_n for $(n \geq 0)$ are defined recursively as follows: $F_0 = 0, F_1 = 1$, and for $n \geq 2, F_n = F_{n-1} + F_{n-2}$.
   Proof by induction on n that $\sum_{i=0}^{n} F_i = F_{n+2} - 1$.

   (b) Solve the following recurrence by applying the Master Theorem, and then show by induction that the result is correct.

   $$T(1) = 4 \text{ and for all } n \geq 2 \text{ a power of 2, } T(n) = 2T(n/2) + 3n + 2.$$

2. (5) **True or False?**

   (a) $\log n \in O(\sqrt{n})$.
   (b) $n \log n \in O(\sqrt{n})$.
   (c) $\sqrt{n} \log n \in O(n)$.
   (d) $n^2/\log n \in o(n^2)$.
   (e) $2^n \in \Theta(n!)$.

3. (10) Consider a stack of n pancakes of distinct diameters. We want to sort the pancakes so that they are increasing in diameter from the top to the bottom of the stack using only "flip" operations. To "flip" the top k pancakes, put a spatula under the kth pancake and invert the stack above that point. For example, if the sizes of the pancakes are 1,5,3,8,2,7 from top to bottom, flipping the top 4 pancakes would give 8,3,5,1,2,7.

   Describe an algorithm to sort the pancakes using at most $2n - 3$ flips, for $n > 1$. (you do not need to express the algorithm in pseudo-code if your English description is clear.) Explain why your algorithm uses at most $2n - 3$ flips.

4. (10) You are given two 2-3-4 trees T and T', storing m and n keys respectively. Every key stored in T is smaller than every key stored in T'. How can you merge the two trees into a 2-3-4 tree storing $m + n$ keys using only $O(\log m + \log n)$ time?

5. (15) The matrix product problem can described as follows: Given n matrices $A_1, A_2, \ldots, A_n$, where for $1 \leq i \leq n$, A_i is a $p_{i-1} \times p_i$ matrix, parenthesize the product $A_1.A_2.\ldots A_n$ so as to minimize the total cost, assuming that the cost of multiplying an $p_{i-1} \times p_i$ matrix by a $p_i \times p_{i+1}$ matrix is $p_{i-1}p_ip_{i+1}$.

   (a) Let $m[i, j]$ denote the cost of the optimal matrix chain product of $A_i.A_{i+1}.\ldots A_j$. Describe the recursive formula to compute $m[i, j]$.

7

(b) Fill the cost table m in the dynamic programming algorithm for iterated matrix products on the inputs: $n = 4; p_0 = 2, p_1 = 5, p_2 = 4, p_3 = 1, p_4 = 10$.

(c) Draw the recursion tree for the problem in (b). Indicate the nodes of the recursion tree that are evaluated in the Memoization process.

6. (10) Euro coin denominations are $1, 2, 5, 10, 20, 50, 100, 200$. Determine if the greedy approach for the coin changing problem with Euro deniminations is optimal. If it is not optimal, then give a counterexample.

7. (10) It is easy to show that the greedy algorithm is not optimal when the denomination set is $\{1, 8, 12\}$ (for the change of $t = 16$, the optimal solution is 2×8). Let $C(t)$ denote the optimal solution for the coin change problem.

   (a) Give a recursive formula for $C(t)$.

   (b) Draw the recursion tree for $C(16)$.

   (c) Identify the subproblems that the dynamic programming approach will solve to compute $C(16)$.

8. ~~(5) Find the minimum and the maximum number of edges that may be considered by the Kruskal's algorithm when given a connected weighted graph with n vertices.~~

9. (10) Define a $1 - tree$ of an undirected weighted graph to be the spanning tree with one extra edge added. Design and analyze an algorithm to find a minimum 1-tree of a weighted undirected graph. Justify its correctness.

10. ~~(15) Consider a set $V = \{p_1, p_2, \ldots, p_n\}$ of n points in the plane. Consider a graph $G = (V, E)$ where $E = \{(p_i, p_j) | \text{ for all } 1 \le i < j \le n\}$. The weight of each edge (p_i, p_j) is the distance between p_i and p_j.~~

    ~~(a) Why is the cost of a minimum spanning tree a lower bound on the cost of the traveling salesman tour in G?~~

    ~~(b) Describe either Prim's or Kruskal's algorithm to compute a minimum spanning tree of G. Analyze your algorithm.~~

    ~~(c) How can a minimum spanning tree be used to obtain a traveling salesman tour whose cost is at most twice that of an optimal tour?~~

    ~~(d) Give a short description of the dynamic programming approach to compute the optimal tour.~~

11. (10) Given a directed graph $G = (V, E)$ where every vertex u has weight w_u and every edge (u, v) has weight $W_{(u,v)}$. Define the length of a path $p = v_1 v_2 \ldots v_r$ to be $\sum_{i=1}^{r} w_{v_i} + \sum_{i=1}^{r-1} W_{(v_i, v_{i+1})}$.

    Describe an efficient algorithm for the single-source shortest path problem using this definition. (Hint: This can be easy if you think about transforming the graph rather than writing code.)

Problem 1:
Design an efficient algorithm to find a spanning tree for a connected, weighted, undirected graph G=(V,E) such that the weight of the maximum-weight edge in the spanning tree is minimized.

Problem 2:
Professor Uriah has developed a hardware priority queue for his computer. The priority queue device can store up to *p* records, each consisting of a key and a small amount of satellite date (such as a pointer). The computer to which it is attached can perform Insert and Extract-Min operations on the priority queue, each of which takes O(1) time, no matter how many records are stored in the device. The professor wishes to use the hardware priority queue to help implement a sorting algorithm on his computer. He has n records stored in the primary memory of his machine. If $n \leq p$, the professor can certainly sort the keys in O(n) time by first inserting them into the priority queue, and then repeatedly extracting the minimum. Design an efficient algorithm for sorting $n > p$ items using the hardware priority queue. Analyze your algorithm in terms of both n and p.

Problem 3:
A ski rental agency has m pairs of skis, where the height of the *i*th pair of skis is s_i. There are n skiers who wish to rent skis, where the height of the *i*th skier is h_i. Ideally, each skier should obtain a pair of skis whose height matches his own height as closely as possible. Design an efficient algorithm to assign skis to skiers so that the sum of the absolute differences of the heights of each skier and his skis is minimized. (For partial credit, assume m = n)

Problem 4:
A directed graph G = (V, E) is unipathic if for any two vertices $u, v \in V$, there is at most one simple path from u to v. Suppose a unipathic graph G has both positive and negative edge weights. Design an efficient algorithm to determine the shortest-path weights from a source s to all vertices $v \in V$ for such a unipathic graph. If some shortest-path weights to vertices reachable from s do not exist, your algorithm should report that a negative-weight cycle exists in the graph.

Problem 5:
Show how to implement a dynamic set that efficiently supports the FIFO queue operations Enqueue and Dequeue, as well as Minimum.

Problem 6:
A unimodal sequence is a sequence $<a_0, a_1, \ldots, a_{n-1}>$ for which there exists a *t* such that $<a_t, a_{t+1}, \ldots, a_{t+n-1}>$ strictly increases and then strictly decreases, where the subscript calculations are performed modulo n. That is, if the sequence $< a_0, a_1, \ldots, a_{n-1}>$ is rotated to the left t positions, it strictly increases to a maximum and then strictly decreases. Design an efficient algorithm to find the maximum value in a unimodal sequence. (For partial credit, consider only unimodal for which t=0.)

Problem 7:

9

The colonel Motors Corporation of Frankfort, Kentucky has produced a new line of vehicles which require chicken droppings for fuel. Because of this unusual fuel requirement, there are only certain fueling stations in the country where the vehicles can be refilled. Thus, to get from one place to another, an owner must plan a route that ensures that he can get refills along the way. The Colonel Motors Corporation has hired Professor Sanders of the Kentucky Institute of Technology as a consultant to prepare an on-line route-finding service. To use the computersized service, an owner will enter the driving range of his vehicle (the distance the vehicle can go on a single fill-up), the "distance to empty" (the distance the vehicle can to with the fuel that's now in the tank), his initial location (source), and his desired destination. The service will either respond with a shortest route from the source to the destination such that the owner never runs out of fuel, or it will deem that no such route exists. Model the professor's problem in terms of a weighted, directed graph in which streets are edges, intersections are vertices, and some intersections have fueling stations, and design an efficient algorithm to solve it. Assume that a driver's source and destination are also at intersections. (For partial credit, design an algorithm to find any feasible path form the source to the destination.)

Problem 8:
Professor Babylonia wants to construct the tallest tower possible out of building blocks. She has n types of blocks, and an unlimited supply of blocks of each type. Each type-*i* block is a rectangular solid with linear dimensions (x_i, y_i, z_i). A block can be oriented so that any two of it's three dimensions determine the dimensions of a base and the other dimension is the height. In building a tower, one block may be ~~placed~~ on top of another block as long as the two dimensions of the base of the upper block are each ~~strictly~~ smaller than the corresponding base dimensions of the lower block. (Thus, for example, blocks oriented to have equal-sized bases *cannot* be stacked.) Design an efficient algorithm to determine the tallest tower that the professor can build. (For partial credit, assume that the blocks cannot be reoriented.)

Problem 9:
Give asymptotically tight upper (big O) bounds for T(n) in each of the following recurrences, Justify your solutions by naming the particular case of the master theorem, by iterating the recurrence, or by using the substitution method.
a. $T(n) = T(n-2) + 1$
b. $T(n) = 2T(n/2) + n \lg^2 n$
c. $T(n) = 9T(n/4) + n^2$
d. $T(n) = 3T(n/2) + n$
e. $T(n) = T(n/2) + \sqrt{n}$

Problem 10:
Consider the code for Build-Heap, which operations on a heap stored in an array A[1...n]:

```
Build-Heap(A)
1 heap-size[A] ← length[A]
2 for i ← ⌊length[A]/2⌋ downto 1
3       do Heapify(A,i)
```

a. show this procedure can be implemented as a recursive divide-and-conquer procedure Build-Heap(A,i), where A[i] is the root of the subheap to be built. To build the entire heap, we would call Build-Heap(A,1).
b. Give a recurrence that describes the worst-case running time of your procedure.
c. Solve the recurrence using the master method (~~or otherwise for period credit~~).

~~**Problem 11:**~~
~~Let G = (V,E) be a flow network with source s, sink t, and suppose each edge e ∈ E has capacity c(e) =1. Assume also, for convenience, that $|E| = \Omega(V)$.~~

1. ~~Suppose we implement the Ford-Fulkerson maximum-flow algorithm by using depth-first search to find augmenting paths in the residual graph. What is the worst-case running time of this algorithm on G?~~
2. ~~Suppose a maximum flow for G has been computed, and a new edge with unit capacity is added to E. Describe how the maximum flow can be efficiently updated. (Note: It is not the value of the flow that must be updated, but the flow itself.) Analyze your algorithm.~~
3. ~~Suppose a maximum flow for G has been computed, but an edge is now removed from E. Describe how the maximum flow can be efficiently updated. Analyze your algorithm.~~

Problem 13:
Give a short answer for each problem below.
a. Prove that $(n+1)^2 = O(n^2)$ by giving the constants n_0 and c in the definition of O-notation. Justify your answer.
b. Suppose that you want to sort n numbers, each of which is either 0 or 1. Briefly describe an asymptotically efficient algorithm for this problem. What is its running time?
c. Briefly describe what we mean by a randomized algorithm, and name two examples.
d. Consider a priority queue that supports the operations Insert and Extract-Min. Argue that in the worst case, if we perform a mixture of n Insert and Extract-Min operations on priority queue, there is at least one operation that takes $\Omega(\lg n)$ time. Assume that the order of elements in the priority queue is determined by comparisons only. (hint: For any set of numbers, at lest one number must equal or exceed the average.)

Problem 14:
Consider a ser S of $n \geq 2$ distinct numbers given in unsorted order. Each of the following four problem parts asks you to give an algorithm to determine two distinct numbers x and y in the set S that satisfy a stated condition. In as few words as possible, describe your algorithm and justify its running time. To keep your answers brief, use algorithms from lectures and the book as subroutines.
a. In O(n) time, determine $x, y \in S$ such that

$$|x - y| \geq |w - z|$$

for all $w, z \in S$

b. In O(n lg n) time, determine $x, y \in S$ such that $x \neq y$ and

$$|x - y| \leq |w - z|$$

for all $w, z \in S$ such that $w \neq z$.

~~c. In O(n) expected time, determine x, y ∈ S such that~~

~~$x+y = Z$~~

~~where Z is given, or determine that no such numbers exist. (hint: $y = Z - x$.)~~

d. In O(n) time determine x , y ∈ S such that

$$|x - y| \le \frac{1}{n-1}\left(\max_{z\in S} z - \min_{z\in S} z\right),$$

i.e., determine any two numbers that are at least as close together as the average distance between consecutive numbers in the sorted order.~~(Hint: Use divide and conquer.)~~