The Hong Kong University of Science & Technology
Department of Computer Science

COMP271: Design & Analysis of Algorithms

Spring 2000

Mid-term Examination

Date: 18:00-20:00pm, April 20, 2000
Room 2302

Problem 1

Please justify if the following statements are true. You need not to give the reasons.

(1) If $f_1 = (n+3)^3$, $f_2 = 2n^3 + n^2 \log n$, then
$f_1 = O(f_2)$, $f_1 = \Omega(f_2)$ and $f_1 = \Theta(f_2)$

Answer:True

(2) We can sort any six integers by 9 comparisons.

Answer:False
A decision tree of 9 comparisons can only distinguish at most 512 kinds of permutation of six integers, but the number of possible permutation is 6!=720, so we can't sort them by only 9 comparisons.

(3) A graph may have many minimum spanning trees.
Answer: True

Problem 2

(a) Explain why we can not use master method to solve recurrence equation
$T(n) = 2T(n/2) + n \log n$

Answer:
In master method, for this case, $n^{\log_b a}$ is n, f(n) is *n*log*n*, Obviously we can not apply case 1 and case 2 on it.
If we apply case 3, we need to find an $\varepsilon > 0$, *let* $n \log n = \Omega(n^{1+\varepsilon})$, that's impossible.
So we can't apply any of the three cases, thus can't use master method on the recurrence equation.

(b) Solve the following recurrence equation:

$$T(n) = \begin{cases} 1, & if\ n = 1,2 \\ 3T(n/3) + n, & if\ n > 2 \end{cases}$$

You can use any method you wish.

Answer:

Use master method, $n^{\log_b a}$ is n, f(n) is *n*, there are the same, so we can apply case 2 on it, and directly get the answer $T(n) = \Theta(n \log n)$.

We can also use iteration method, the result is the same. But some students get a result as $T(n) = \Theta(n + n \log n)$, or $\Theta(n \log_3 n)$, in fact, that's just $T(n) = \Theta(n \log n)$.

<u>Problem 3</u>

MAXMIN problem is to find the maximum element and the minimum element from *n* integers. The following is the pseudo code for the MAXMIN problem by using the divide & conquer method. Please evaluate the time complexity of this algorithm.

```
MAXMIN(L, max, min)
If |L|=1 then begin
        max:= the only element in L;
        min:=max;
end
else begin
        divide L into two parts L1 and L2 in the middle;
        MAXMIN(L1,max1, min1);
        MAXMIN(L1,max2, min2);
        max:=max(max1,max2);
        min:=min(min1,min2);
end;
```

Answer:

In the pseudo code, all codes except the recursive function call cost only constant time, So we can built a recurrence equation as T(n)=2T(n/2)+O(1). Use master method, $n^{\log_b a}$ is n, f(n) is $O(1)$, so we can apply case 1, the result is O(n).

Problem 4 The following pseudo code outputs the prefix expression of the expression represented by syntax tree G. Revise the code to transform the output sequence to the infix expression. Note that in infix expression, we need add brackets in the expression, while in postfix and prefix expression we do not need add brackets. A sample output of infix expression is (3+((5-3)*(4-2)).

```
Prefix(G, u){
        Output the symbol stored in u;
        if  ( left_child(u) != NIL )
                Prefix(G, left_child(u) );
        if  ( right_child(u) != NIL )
                Prefix(G, right_child(u) );
}
```

Answer:

```
Infix(G, u){
        if ( left_child(u)==NIL || right_child(u) ==NIL ) {
                Output the symbol stored in u;
                return;
        }
        Output '(';
        if  ( left_child(u) != NIL )
                Prefix(G, left_child(u) );
        Output the symbol stored in u;
        if  ( right_child(u) != NIL )
                Prefix(G, right_child(u) );
        Output ')';
}
```

Some students forgot to add brackets, the calculation result of the expression will not be the same as the syntax tree. Some others' code will add a pair of brackets for even a single number, but that's OK, because the result of the expression is correct.

Problem 5

(a) List six real problems that can be represented by graph.

See lecture notes, there are many such kind of problems.

(b) Traverse the following directed graph $G = (V, E)$ by using Depth-First Search starting from vertex a. You must use the adjacent list representation given below. Show the tree edges, back edges and forward edges produced by the search.

$a :\rightarrow b, c$

$b :\rightarrow d, e$

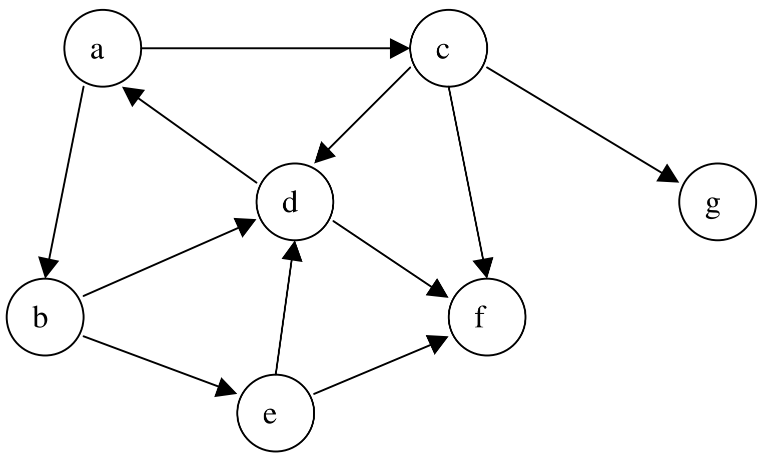
$c :\rightarrow d, f, g$

$d :\rightarrow a, f$

$e :\rightarrow d, f$

$f :\rightarrow \phi(empty)$

$g :\rightarrow \phi(empty)$



Answer:

The visiting sequences is a b d f e c g. So a→b, b→d, d→f, b→e, a→c, c→g is tree edges. There is only one back edge d→a, and no forward edge. See lecture notes about the definition of back edge and forward edge.