University of Bergen
Faculty of Mathematics and Natural Sciences
Examination in course INF102 - Algorithms, Data Structures and Programming
Friday, December 10, 2004, 09:00 - 14:00
English text

No books or notes are allowed.

Each algorithm and pseudo code that you give, must be explained and justified either with ordinary text or, if you find it necessary, with a formal proof.

If you are asked to give the running time, or asked to produce an algorithm with a given running time, you should explain how you obtain this running time.

1 Complexity and Recursion

1. Using the Big Oh notation, indicate the time requirement of each of the following methods.

```
1) public static void F1(int n)
{
     int sum=0;
     for (int i=0; i<n; i++)
       for (int j=0; j<i; j++)
         sum++;
}

2) public static int F2(int n)
{
     if (n<=1) return 1;
  else return 3*F2(n/3);
}

3) public static int F3(int n)
{
     if (n<=1) return 1;
  else return F3(n-2)+F3(n-2);
}

4) public static int F4(int n)
{
     if (n<=1) return 1;
  else return F4(n-1)+F4(n-2);
}

5) public static void F5(int n)
{
```

```
     int sum=0;
     for (int i=0; i<2*n; i++)
       for (int j=0; j<2*i; j++)
        for (int k=0; k<2*j; k++)
         sum=sum+k;
}
```

```
6)public static int F6(int n)
{
     if (n<=1) return 0;
  else return 2*F6(n/2)+n;
}
```

2. What is the output of the following method?

```
public static void main(String[] args)
    {
   int i=5;
 System.out.println(  F3(F2(i))+F6(i/2));
   }
```

2 Sorting

1. Write a pseudocode describing a recursive insertion sort. What is the best and the worst case running time of the algorithm? Explain your answer.

2. Sort the array of integers 5 6 1 9 8 12 0 2 10 into ascending by using an insertion sort. Write the contents of the array each time that the sort changes it.

3. Sort the array of integers 5 6 1 9 8 12 0 2 10 into ascending by using the Shell sort with index separations 4, 2, and 1. Write the contents of the array each time that the sort changes it.

4. Sort the array of integers 5 6 1 9 8 12 0 2 10 into ascending by using a quick sort. Use median-of-three pivot selection. Write the contents of the array each time that the sort changes it.

3 Trees

1. Consider a binary search tree as the one pictured in the left half of Fig. 1.Now imagine that you traverse this tree and save its data in a file. Show the results of adding the search keys into an initially empty binary search tree from the obtained file if the tree was traversed in **a.) preoder b.) inorder c.) level order d.) postorder**.

2. Trace the steps of a heapsort on the following array 5 12 9 81 4 2 10 11 13 1.

3. Draw a tree that results when you add the values 9, 20, 69, 48, 48, 60, 70, 75, 1, and 100 to **a.) AVL tree b.) A red-black tree**.

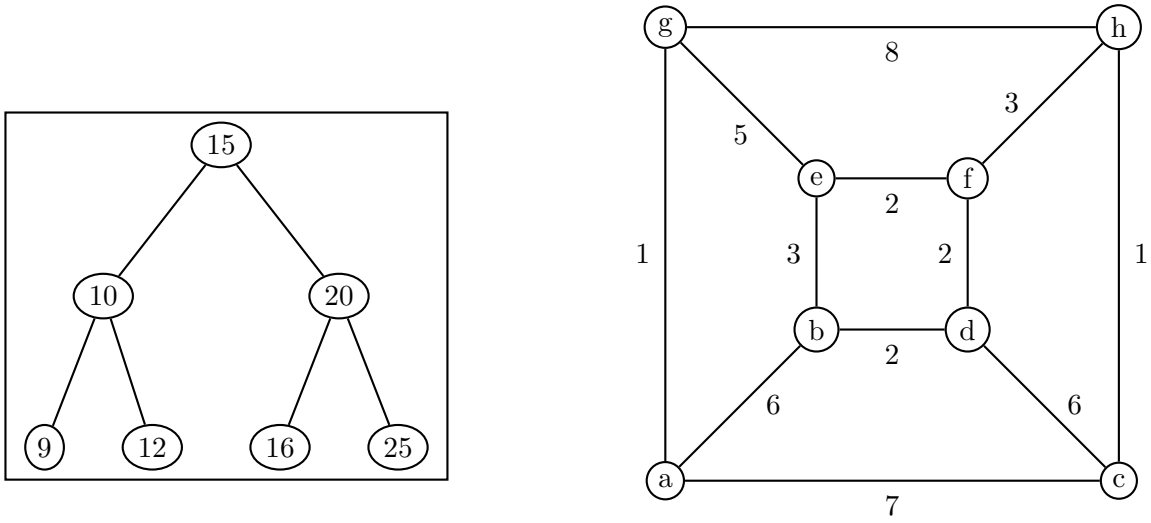

Figure 1: Binary search tree and weighted graph

4 Graphs

1. Write a pseudo code for an algorithm that for a given directed graph G on n vertices and m edges determines in time $\mathcal{O}(n+m)$ if G has a cycle. Don't forget to explain why the running time of your algorithm is $\mathcal{O}(n+m)$.

2. Consider the weighted graph as pictured in the right half of Fig. 1. Trace the steps of the following algorithms

   a) Based on breadth-first traversal algorithm finding the cheapest path from a to h.

   b) Prim's algorithm computing minimum spanning tree.

   b) Kruskal's algorithm computing minimum spanning tree.

5 Algorithm design

In chess, a queen can move as far as she pleases, horizontally, vertically, or diagonally. A chess board has 8 rows and 8 columns. The Queen's problem asks how to place 8 queens on a chess board so that none of them can hit any other in one move. Write a pseudo code of an algorithm solving this problem.

Fedor Fomin Pinar Heggernes

Solutions

Complexity & Recursion

$T(n)$ running time. A)
1) $\sum_{i=1}^{n} \sum_{j1}^{i} j = (1) + (1+2) + (1+2+3) + \ldots + (1+2+\ldots+n)$. The number of summations is $(n-1) + 1 + 2 + \ldots + n - 1$. ANSWER: $T(n) = \mathcal{O}(n^2)$.

2) $T(n) = 3T(n/3)$. $T(n) = \log_3 n$. ANSWER: $T(n) = \mathcal{O}(\log n)$.

3) $T(n) = 2T(n-2)$. $T(n) = \sqrt{2}^n$. ANSWER: $T(n) = \mathcal{O}(\sqrt{2}^n)$.

4) $T(n) = T(n-1) + T(n-2)$. We are looking for $T(n) = \alpha^n$. $\alpha^n = \alpha^{n-1} + \alpha^{n-2}$. Thus $\alpha^2 = \alpha + 1$. ANSWER: $T(n) = \mathcal{O}((\frac{1+\sqrt{5}}{2})^n)$.

5) ANSWER: $T(n) = \mathcal{O}(n^3)$.

6) $T(n) = 2 * T(n/2) + n$. $T(1) = 0$. By induction, $T(n) = n \log_2 n = \mathcal{O}(n \log)$.
B) 4

Sorting

1)

```
Algorithm insertionSort(a, first, last)
// Sorts the array elements a[first] through a[last] recursively.

if (the array contains more than one element)
{
   Sort the array elements a[first] through a[last-1]
   Insert the last element a[last] into its correct sorted position
   within the rest of the array
  }
```

Best running time: $\mathcal{O}(n)$. Worst running time: $\mathcal{O}(n^2)$.
2)

```
5 6 1 9 8 12 0 2 10
1 5 6 9 8 12 0 2 10
1 5 6 8 9 12 0 2 10
0 1 5 6 8 9 12 2 10
0 1 2 5 6 8 9 12 10
0 1 2 5 6 8 9 10 12
```

3)

```
Separation 4

-       -         -
5 6 1 9 8 12 0 2 10

  -       -
5 6 1 9 8 12 0 2 10

    -         -
5 6 1 9 8 12 0 2 10

      -         -
5 6 0 9 8 12 1 2 10
```

```
      _         _
5 6 0 2 8 12 1 9 10

        _       _
5 6 0 2 8 12 1 9 10
```

Separation 2

```
_   _   _    _    _
5 6 0 2 8 12 1 9 10

_   _   _    _    _
0 6 5 2 8 12 1 9 10

_   _   _    _    _
0 6 1 2 5 12 8 9 10

  _   _   _    _
0 6 1 2 5 12 8 9 10

  _   _   _    _
0 2 1 6 5 12 8 9 10

  _   _   _    _
0 2 1 6 5 9 8 12 10
```

Separation 1

```
0 2 1 6 5 9 8 12 10
0 1 2 6 5 9 8 12 10
0 1 2 5 6 9 8 12 10
0 1 2 5 6 8 9 12 10
0 1 2 5 6 8 9 10 12
```

3)

```
_       _       _
5 6 1 9 8 12 0 2 10   //Choosing pivot

_       _       _
8 6 1 9 5 12 0 2 10   //5 is chosen as pivot

8 6 1 9 10 12 0 2 5   //pivot is swapped with the last

2 6 1 9 10 12 0 8 5

2 0 1 9 10 12 6 8 5

2 0 1 5 9 10 12 6 8 //pivot is placed on its position

2 0 1 5 9 10 12 6 8 //pivot is placed on its position
//call quick sort recursively for 2 0 1 and 10 12 6 8

1 2 0 5 9 10 12 8 6 //pivots 0 and 6
0 1 2 5 6 9 10 12 8 // recursion for 9 10 12 8
0 1 2 5 6 8 10 12 9 // pivot 9
0 1 2 5 6 8 9  10 12
```

Trees

Graphs

1. There are several solutions for this problem. For example, we can use the $\mathcal{O}(m+n)$ algorithm for finding topological sort. For each connected component we run the following

```
Algorithm getTopologicalSort()
//vertexStack = a new stack to hold vertices as they are visited
//n = number of vertices in the graph
       for (counter = 1 to n)
          {  nextVertex = an unvisited vertex whose neighbors,
             if any, are all visited
             Mark nextVertex as visited
             stack.push(nextVertex)
          }
   return stack
```

   What remains is to process the stack and check if what we have in stack is topological ordering or no, which means that there all edges go from top to down vertices. If the result is topological ordering, there is no cycle, if no there is a cycle. Thus

```
i=1
  while (stack is not empty)
//label vertices in topological ordering
{  Label stack.pop() by i
  i++
 }
       for (i = 1 to n)
{Check if there is an edge from vertex
labelled i to vertex with smaller label.
If there is such an edge, the graph has cycle
}
```

Algorithm design

The program finds solutions by starting with a queen in the top left corner of the chess board. It then places a queen in the second column and moves it until it finds a place where it cannot be hit by the queen in the first column. It then places a queen in the third column and moves it until it cannot be hit by either of the first two queens. Then it continues this process with the remaining columns. If there is no place for a queen in the current column the program goes back to the preceding column and moves the queen in that column. If the queen there is at the end of the column it removes that queen as well and goes to the preceding column. If the current column is the last column and a safe place has been found for the last queen, then a solution of the puzzle has been found. If the current column is the first column and its queen is being moved off the board then all possible configurations have been examined, all solutions have been found, and the algorithm terminates.