224

B.Sc. Engg. / HD CSE 4th Semester (51) **23 September 2012 (Morning)**

ISLAMIC UNIVERSITY OF TECHNOLOGY (IUT)

ORGANISATION OF ISLAMIC COOPERATION (OIC)

Department of Computer Science and Engineering (CSE)

SEMESTER FINAL EXAMINATION **SUMMER SEMESTER, 2011-2012**

DURATION: 3 Hours **FULL MARKS: 150**

CSE 4403: Algorithms

Programmable calculators are not allowed. Do not write anything on the question paper.

There are **8 (eight)** questions. Answer any **6 (six)** of them.

Figures in the right margin indicate marks.

---

1. a) The algorithm for matrix-chain multiplication problem is as follows: 13

```
Matrix-Chain-Order(p)
n = p.length-1
let m[1..n, 1..n] and s[1..n-1, 2..n] be new tables
for i = 1 to n
     m[i, i] = 0
for l = 2 to n        //  l is the chain length
     for i = 1 to n-l+1
          j  = i+l-1
          m[i, j] = ∞
          for k = i to j-1
               q = m[i, k] + m[k+1, j] + p[i-1] p[k] p[j]
               if  q < m[i, j]
                    m[i, j] = q
                    s[i, j] = k
return m and s
```

Find an optimal parenthesization of a matrix-chain product whose sequence of dimensions is <5, 10, 3, 12, 5>.

b) Find the structure of an optimal parenthesization and a recursive solution to the matrix-chain multiplication problem. 12

2. a) In which situation dynamic programming is preferred over divide-and-conquer approach? 6

b) Modify the pseudocode provided below to reconstruct an LCS from the completed c table and the original sequence $X = <x_1, x_2, \ldots, x_m>$ and $Y = <y_1, y_2, \ldots, y_n>$ in $O(m+n)$ time, without using the b table. 12

```
Print-LCS(b, X, i, j)
if i == 0 or j == 0
     return
if b[i, j] ==  "↖"
     Print-LCS(b, X, i-1, j-1)
     print $x_i$
elseif b[i, j] ==  "↑"
     Print-LCS(b, X, i-1, j)
else Print-LCS(b, X, i , j-1)
```

c) In activity selection problem, consider any nonempty subproblem S_k, and let a_m be an activity in S_k with the earliest finish time. Prove that a_m is included in some maximum-size subset of mutually compatible activities of S_k. 7

3. a) Discuss how both the 0-1 and fractional knapsack problem exhibit the optimal substructure property. 8

   b) In dynamic programming we make a choice at each step, but the choice usually depends on the solutions to subproblems. In other words we must solve the subproblems before making a choice in case of dynamic programming. How does greedy strategy differ in this aspect and how does it work? 7

   c) Discuss the disjoint-set operations. With the help of an example show how disjoint-set data structure is implemented using linked list. Your discussion should include the weighted-union heuristic that is used in the linked-list representation of disjoint sets. 3+4+3

4. a) With the help of an example show that an n-element heap has height $\lfloor \lg n \rfloor$. 6

   b) Use an example to show that there are at most $\left\lceil \frac{n}{2^{h+1}} \right\rceil$ nodes of height h in any n-element heap. Afterwards show that the cost of Build-Max-Heap procedure is $\sum_{h=0}^{\lfloor \lg n \rfloor} \left\lceil \frac{n}{2^{h+1}} \right\rceil O(h) = O(n)$ 4+8

   c) How can the max-priority queue be used in job scheduling and the min-priority queue be used in an event-driven simulator? 7

5. a) Suppose the array A = <15, 14, 10, 8, 7, 9, 3, 2, 4, 1> represents a heap that maintains the max-heap property. Show the steps when the procedure Heap-Increase-Key (A, 9, 16) is called. 10

```
Heap-Increase-Key(A, i, key)
    if key < A[i]
        error "new key is smaller than current key"
    A[i] = key
    while i > 1 and A[Parent(i)] < A[i]
        exchange A[i] with A[Parent(i)]
        i = Parent(i)
```

   b) Step by step, show the execution of *Bellman-Ford's algorithm* on the graph provided in figure 1 (node *a* is the source *s*). Use the following sequence to process the edges: (a, f), (a, b), (a,c), (b, c), (b, d), (c, f), (c, d), (d, e), (f, e) 15

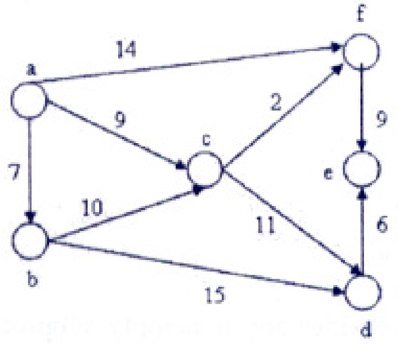

Figure 1

226

```
Bellman-Ford(G, w, s)
    Initialize-Single-Source(G, s)
    for i = 1 to |G.V|-1
        for each edge (u, v) ∈ G.E
            Relax(u, v, w)
        for each edge (u, v) ∈ G.E
            if v.d > u.d + w(u, v)
                return FALSE
    return TRUE
```

6. a) What is meant by a cut respects a set A of edges? Justify with an example. 2+3

   b) Show step by step execution of *Prim's algorithm* on the graph provided in figure 2 (the root vertex is *a*). 15

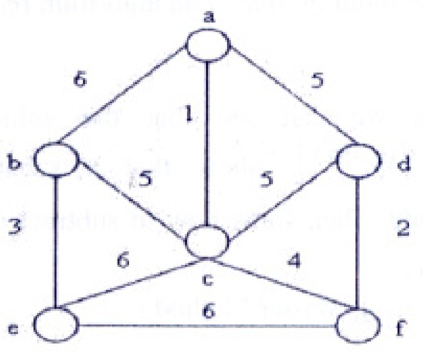

Figure 2

```
Mst-Prim (G, w, r)
    for each u ∈ G.V
        u.key = ∞
        u.Π = NIL
    r.key = 0
    Q = G.V
    while Q ≠ ∅
        u = Extract-Min(Q)
        for each v ∈ G.Adj[u]
            if v ∈ Q and w(u, v) < v.key
                v.Π = u
                v.key = w(u, v)
```

   c) Provide a simple analysis on the running time of *Prim's algorithm*. 5

7. a) Solve the recurrence $T(n) = T(n-1) + n$ using substitution method. As you already know that in substitution method you need to guess the solution at first before proving it, your task is to guess the solution of the recurrence using recursion tree and later prove it using substitution method. 10

b) The following brute-force algorithm solves the maximum-subarray problem in $O(n^3)$ time. Modify the algorithm to solve the same problem in a brute-force manner in $O(n^2)$ time. 10

```
Brute-Force-Max-Subarray (A)
vmax = A[1]
n = A.length
for i = 1 to n
    for j = i to n
        //calculate v(i, j)
        v = 0
        for x = i to j
            v = v + A[x]
        if v > vmax
            vmax = v
return vmax
```

c) What does order of growth of the running time of an algorithm represent? 5

8. a) By applying Master theorem we can see that the solution to the recurrence $T(n) = 4T(n/3) + n$ is $T(n) = \Theta(n^{\log_3 4})$. Show that a substitution proof with the assumption $T(n) = \Theta(n^{\log_3 4})$ fails. Then show how to subtract off a lower-order term to make the substitution proof work. 15

b) Solve the following recurrences using Master Method : 5+5

i. $T(n) = 3T(\sqrt{n}) + \lg n$

ii. $T(n) = 3T(n/3) + n/\lg n$

Note:
*If a recurrence is applicable to case 3 of master method then you also need to show that the regularity condition holds.
*If Master theorem cannot be used to solve a recurrence then you need to show the reason.