يوم الامتحان: الاربعاء

المستوي الرابع (حاسب)

تاريخ الامتحان: 12 / 6 / 2019 م

المادة : موضوعات مختارة في علوم الحاسب 2 (455 رس)

الممتحن: د/ مصعب عبد الحميد محمد حسان

مدرس بقسم الرياضيات بكلية العلوم

الاسئلة و نموذج الإجابة

ورقة كاملة

Benha University
Faculty of Science
Dept. of Mathematics

Time: Two Hours
Second Semester 2018-2019
Date : 12/6/2019

Selected Topics in Computer Science (2) (MC455) for Fourth Level Students (Computer Science)

Answer the following questions:

Question 1. (20 marks)

A-Define subgraph isomorphism, complete graph, and graph invariant. (4 marks)

B-Discuss Ullman algorithm for graph isomorphism and subgraph isomorphism in details. (9 marks)

C-Compare between adjacency list representation and adjacency matrix representation of graphs. (7 marks)

Question 2. (20 marks)

A- Write the MaxMin algorithm. (7 marks)

B- Solve the following recurrence equation. (6 marks)

$$T(n) = T(n - 1) + n - 1 \quad \text{for } n > 1$$
$$= 0 \quad \text{for } n = 1$$

C- Write the dynamic programming algorithm of Fibonacci numbers. (7 marks)

Question 3. (8 marks)

Choose the correct answer for each of the following

a- A vertex is said to be an isolated vertex if there is ……. incident with it

(A) edge (B) no edge (C) no vertex (D) vertex

b- The overall time taken by MaxMin algorithm is

(A) 2n/3 – 2 (B) 3n/2 – 2 (C) 2n/3 – 3 (D) 3n/2 – 3

c- The overall time taken by mergesort algorithm is

(A) O(n log n) (B) O(log n) (C) O(n^2 log n)

d- An algorithm which uses the past results and uses them to find the new results is…..

(A) Brute-Force (B) Divide and Conquer
(C) Dynamic programming algorithms
(D) None of the mentioned

Best Wishes

Dr. Mosab Abd El-Hameed

Model Answer

Selected Topics in Computer Science (2) (MC455) for Fourth Level Students (Computer Science)

Answer of Question 1

A-

Given two graphs H = (VH, EH, ΣVH, ΣEH, LH) and G = (VG, EG, ΣVG, ΣEG, LG). <u>A subgraph isomorphism</u> from H to G is a injection f : VH $\rightarrow$ VG such that:

1. **(u, v)   EH   iff   (f(u), f(v))   EG,**
2. **LH(u) = LG(f(u)) ∀u   VH, and**
3. **LH((u, v)) = LG((f(u), f(v))).**

A graph G is said to <u>complete</u> (or fully connected or strongly connected) if there is a path from every vertex to every other vertex. A complete graph with *n* vertices will have *n* (*n* – 1)/2 edges

<u>A graph invariant</u> is a function T such that if applied to two isomorphic graphs H and G, then T(H) = T(G). In other words, if T(H) ≠ T(G) then H is not isomorphic to G.

B-

Ullman algorithm is the earliest and highly-cited approach to the (sub)graph isomorphism problem. Given two graphs G1 and G2. To check if G1 is subgraph of G2, Ullman's basic approach is to enumerate all possible mappings of vertices in V_{G1} to those in V_{G2} using a depth-first tree-search algorithm. In order to cope with subgraph isomorphism problem efficiently, Ullman proposed a refinement procedure to prune the search space. It is based on the following three conditions:

1. *Label and degree condition.*

A vertex u ∈ V_{G1} can be mapped to v ∈ V_{G2} under injective mapping f, i.e v = f(u), if

(i) $L_{G1}(u) = L_{G2}(v)$, and

(ii) $\deg_{G1}(u) \geqslant \deg_{G2}(v)$.

2. *One-to-One mapping of vertices condition.*

Once vertex u ∈ V_{G1} is mapped to v ∈ V_{G2}, we cannot map any other vertex in V_{G1} to the vertex v ∈ V_{G2}.

3. *Neighbor condition.*

By this condition Ullman algorithm examines the feasibility of mapping u ∈ V_{G1} to v ∈ V_{G2} by considering the preservation of structural connectivity. If there exist edges connecting u with previously explored vertices of G1 but there are no counterpart edges in G2, the mapping test simply fails.

Considering the graph isomorphism instead of the subgraph isomorphism, the two graphs must have the same number of vertices and the condition 1 is modified as the following :-

1. *Label and degree condition.*
A vertex u ∈ V_{G1} can be mapped to v ∈ V_{G2} under bijective mapping f, i.e v = f(u), if
(i) $L_{G1}(u) = L_{G2}(v)$, and
(ii) $\deg_{G1}(u) = \deg_{G2}(v)$.

C-

Comparison	Winner
Faster to test if (x, y) is in graph?	adjacency matrices
Faster to find the degree of a vertex?	adjacency lists
Less memory on small graphs?	adjacency lists
Less memory on big graphs?	adjacency matrices
Edge insertion or deletion?	adjacency matrices
Faster to traverse the graph?	adjacency lists
Better for most problems?	adjacency lists

Table : Relative advantages of adjacency lists and matrices.

Answer of Question 2

A-

```
MaxMin(S)
 if |S| = 2 then
    Let S = {a, b}
   return (MAX(a, b), Min(a, b))
 else
   Divide S into two subsets S1 and S2 , each with half of the elements
   (max1, min1) = MaxMin(S1)
   (max2, min2) = MaxMin(S2)
 return (MAX(max1, max2), Min(min1, min2))
```

B-

$T(n) = T(n-1) + n - 1$

$= T(n-2) + 2n - 2 - 1$
$= T(n-3) + 3n - 3 - 2 - 1$
$= \dots\dots$
$= T(n-(n-1)) + (n-1)n - (n-1) - \dots\dots - 1$

= T(0) + (n-1)n – (n-1) – ……. – 1

= 0 + (n-1)n - $\sum_{i=0}^{n-1} i$
= n(n-1) – n(n-1)/2
= n(n-1)/2
= O(n^2)

C-

```
fib(n)
 seq = zeros(n)
 seq[1] = seq[2] = 1
 for i from 3 to n
    seq[i] = seq[i-1] + seq[i-2]
 return seq[n-1]
```

Answer of Question 3

a- B
b- B
c- A
d- C