CONFIDENTIAL

HERITAGE INSTITUTE OF TECHNOLOGY

Class test I / II / III Examination 2018 Session : 2017-2018

Discipline : B.Tech (CSE)

Paper Code : CSEN2201 Paper Name : Design & Analysis of Algorithms

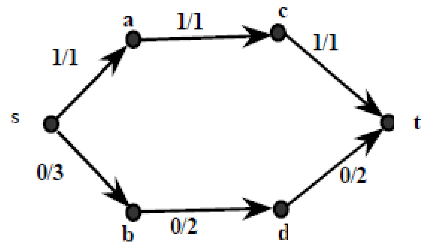
Time Allotted : 1 hr Full Marks : 30

Figures out of the right margin indicate full marks.

Answer all questions.

Candidates are required to give answer in their own words as far as practicable.

Total marks is 36 but the maximum you can score is only 30.

1 (a)	In the algorithm for pattern matching using Finite Automata, the suffix function σ(x) is the _____est _______ of the pattern P that is also a _________ of x. i) large, prefix, suffix ii) small, prefix, suffix iii) large, suffix, prefix iv) None of the above	1x4=4
(b)	Which of the following represents running time of an algorithm to find the MST of a graph G = (V, E) using union-find method i) O(E log E) ii) O(E+V) iii) O (E + VlogV) iv) O(EV)	
(c)	The maximum flow that any augmenting path can accommodate for the flow network shown below is  i) 0.5 ii) 1 iii) 1.5 iv) 2	
(d)	A negative weight cycle can be correctly detected by i) Topological Sorting Algorithm ii) Dijkstra's Algortihm iii) Bellman-Ford Algorithm iv) Prim's Algorithm	
2 (a)	"Shortest path problem follows optimal sub-structure property" – Justify the statement. Answer: ***Lemma 24.1 (Subpaths of shortest paths are shortest paths)*** Given a weighted, directed graph $G = (V, E)$ with weight function $w : E \to \mathbb{R}$, let $p = \langle v_0, v_1, \ldots, v_k \rangle$ be a shortest path from vertex v_0 to vertex v_k and, for any i and j such that $0 \le i \le j \le k$, let $p_{ij} = \langle v_i, v_{i+1}, \ldots, v_j \rangle$ be the subpath of p from vertex v_i to vertex v_j. Then, p_{ij} is a shortest path from v_i to v_j. ***Proof*** If we decompose path p into $v_0 \overset{p_{0i}}{\leadsto} v_i \overset{p_{ij}}{\leadsto} v_j \overset{p_{jk}}{\leadsto} v_k$, then we have that $w(p) = w(p_{0i}) + w(p_{ij}) + w(p_{jk})$. Now, assume that there is a path p'_{ij} from v_i to v_j with weight $w(p'_{ij}) < w(p_{ij})$. Then, $v_0 \overset{p_{0i}}{\leadsto} v_i \overset{p'_{ij}}{\leadsto} v_j \overset{p_{jk}}{\leadsto} v_k$ is a path from v_0 to v_k whose weight $w(p_{0i}) + w(p'_{ij}) + w(p_{jk})$ is less than $w(p)$, which contradicts the assumption that p is a shortest path from v_0 to v_k. ■	(1+2)+(4) + (1+ 3 + 1) = 12

CONFIDENTIAL

HERITAGE INSTITUTE OF TECHNOLOGY

Class test I / II / III Examination 2018 Session : 2017-2018

Discipline : B.Tech (CSE)

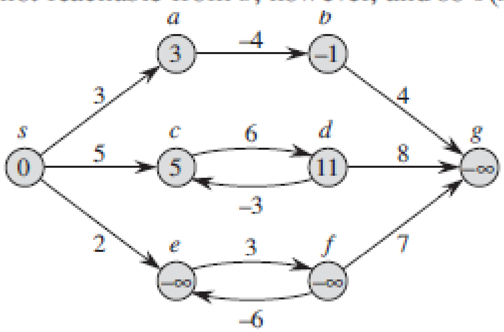
Paper Code : CSEN2201 Paper Name : Design & Analysis of Algorithms

If a graph contains a negative weight cycle reachable from the source vertex then what problem do you face in finding shortest path from that source? Explain with example.

Answer:

If the graph contains a negative-weight cycle reachable from s, however, shortest-path weights are not well defined. No path from s to a vertex on the cycle can be a shortest path—we can always find a path with lower weight by following the proposed "shortest" path and then traversing the negative-weight cycle. If there is a negative weight cycle on some path from s to v, we define $\delta(s, v) = -\infty$.

Because the cycle $\langle e, f, e \rangle$ has weight $3 + (-6) = -3 < 0$, however, there is no shortest path from s to e. By traversing the negative-weight cycle $\langle e, f, e \rangle$ arbitrarily many times, we can find paths from s to e with arbitrarily large negative weights, and so $\delta(s, e) = -\infty$. Similarly, $\delta(s, f) = -\infty$. Because g is reachable from f, we can also find paths with arbitrarily large negative weights from s to g, and so $\delta(s, g) = -\infty$. Vertices h, i, and j also form a negative-weight cycle. They are not reachable from s, however, and so $\delta(s, h) = \delta(s, i) = \delta(s, j) = \infty$.



Note. You can always draw a smaller graph for example.

(b) Give the pseudo-code for Kruskal's algorithm for MST with a very brief explanation of how it works. Note that you do NOT need to write the implementation details of disjoint-set data structure.

CONFIDENTIAL

HERITAGE INSTITUTE OF TECHNOLOGY

Class test I / II / III Examination 2018 Session : 2017-2018

Discipline : B.Tech (CSE)

Paper Code : CSEN2201 Paper Name : Design & Analysis of Algorithms

- **MST-KRUSKAL (G, W)**

1. $A \leftarrow \Phi$
2. for each vertex $v \in V[G]$
3. do MAKE-SET(V)
4. Sort the edges of E by non-decreasing weight w
5. For each edge $(u,v) \in E$, in order by non-decreasing weight
6. do if FIND-SET(u) ≠ FIND-SET(v)
7. then $A \leftarrow A \cup \{(u,v)\}$
8. UNION(u,v)
9. Return A

Lines 1–3 initialize the set A

to the empty set and create |V | trees, one containing each vertex. The **for** loop in lines 5–8 examines edges in order of weight, from lowest to highest. The loop checks, for each edge (u, v) whether the endpoints u and v belong to the same tree. If they do, then the edge (u, v) cannot be added to the forest without creating a cycle, and the edge is discarded. Otherwise, the two vertices belong to different trees. In this case, line 7 adds the edge (u, v) to A, and line 8 merges the vertices in the two trees.

(c) What is the significance of doing topological sort?

Answer:
Topological sort of a DAG G= (V,E) is a linear ordering of all its vertices such that if G contains an edge (u, v) then u appears before v in the ordering. (If graph is not acyclic no linear ordering is possible)

Write the algorithm for topological sorting.

1. call DFS(G) to compute finishing times f[v] for each vertex v.
2. as each vertex is finished, insert it onto the front of a linked list.
3. return the linked list of vertices

Can you apply Topological Sort algorithm on a cyclic graph? Justify your answer.

No, because no linear order of vertices of a cycle, is possible.

CONFIDENTIAL

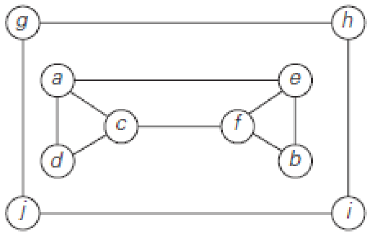
HERITAGE INSTITUTE OF TECHNOLOGY

Class test I / II / III Examination 2018 Session : 2017-2018

Discipline : B.Tech (CSE)

Paper Code : CSEN2201 Paper Name : Design & Analysis of Algorithms

3(a) Consider the following graph: (2 + 2)+ (2 + 2) + (4x0.5) = 10



i) Write the UNION-FIND algorithm to find the connected components of the given graph.

Answer:

```
CONNECTED-COMPONENTS(G)
for each vertex v ∈ G.V
    MAKE-SET(v)
for each edge (u, v) ∈ G.E
    if FIND-SET(u) ≠ FIND-SET(v)
        UNION(u, v)

SAME-COMPONENT(u, v)
if FIND-SET(u) == FIND-SET(v)
    return TRUE
else return FALSE
```

ii) Also show the steps for detailed work-out of the algorithm on this graph.

Edges Processed	Collection of Disjoint sets
initial sets	{g} {h} {i} {j} {a} {e} {b} {f} {c} {d}
(g, h)	{g, h} {i} {j} {a} {e} {b} {f} {c} {d}
(h, i)	{g, h, i} {j} {a} {e} {b} {f} {c} {d}
(i, j)	{g, h, i, j} {a} {e} {b} {f} {c} {d}
(j, g)	{g, h, i, j} {a} {e} {b} {f} {c} {d}
(a, e)	{g, h, i, j} {a, e} {b} {f} {c} {d}
(e, b)	{g, h, i, j} {a, e, b} {f} {c} {d}
(b, f)	{g, h, i, j} {a, e, b, f} {c} {d}
(f, c)	{g, h, i, j} {a, e, b, f, c} {d}
(c,d)	{g, h, i, j} {a, e, b, f, c, d}
(d, a)	{g, h, i, j} {a, e, b, f, c, d}
(a, c)	{g, h, i, j} {a, e, b, f, c, d}

CONFIDENTIAL

HERITAGE INSTITUTE OF TECHNOLOGY

Class test I / II / III Examination 2018 Session : 2017-2018

Discipline : B.Tech (CSE)

Paper Code : CSEN2201 Paper Name : Design & Analysis of Algorithms

(b) Define maximum-flow problem.

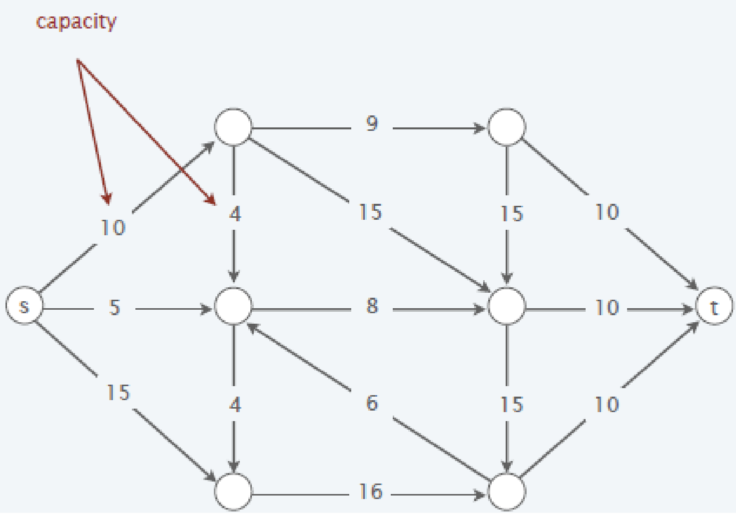
A flow in G is a real valued function f: V x V → R that satisfies some properties. What are they and also state each of them in one sentence.

Answer:

Flow network

- Abstraction for material flowing through the edges.
- Digraph $G = (V, E)$ with source $s \in V$ and sink $t \in V$.
- Nonnegative integer capacity $c(e)$ for each $e \in E$.

no parallel edges
no edge enters s
no edge leaves t



Def. An *st*-flow (flow) f is a function that satisfies:

- For each $e \in E$: $0 \leq f(e) \leq c(e)$ [capacity]
- For each $v \in V - \{s, t\}$: $\sum_{e \text{ in to } v} f(e) = \sum_{e \text{ out of } v} f(e)$ [flow conservation]

Def. The value of a flow f is: $val(f) = \sum_{e \text{ out of } s} f(e)$.

Max-flow problem. Find a flow of maximum value.

Note. You can always draw a smaller graph as an example.

CONFIDENTIAL

HERITAGE INSTITUTE OF TECHNOLOGY

Class test I / II / III Examination 2018 Session : 2017-2018

Discipline : B.Tech (CSE)

Paper Code : CSEN2201 Paper Name : Design & Analysis of Algorithms

(c)	State whether the following problems are NP-Hard or polynomial-time solvable - Set Cover Problem, Edge Cover Problem, Vertex Cover Problem, Eulerian Path Problem. Answer: NP-Hard : Set Cover Problem, Vertex Cover Problem Polynomially solvable: Edge Cover Problem, Eulerian Path Problem	
4.	Define the prefix function Π in the context of KMP pattern matching algorithm. Show how the prefix function works on the pattern P: aababaaba by explaining its methodology, i.e., give the values of Π(a), Π(a), Π(aab), Π(aaba) etc. Answer: Prefix function Π(q) is the length of the longest prefix of pattern P that is proper suffix of P_q Π(q) = max{k: k<q and P_k is a suffix of P_q} Π(a)=0, Π(aa)=1, Π(aab)=0, Π(aaba)=1, Π(aabab)=0, Π(aababa)=1, Π(aababaa)=2, Π(aababaab)=3, Π(aababaaba)=4	(2+4) = 6
5.	A sequence of *n* operations is performed on a data structure. The *i*th operation costs 4 i if *i* is a power of 2, 1 otherwise. Use aggregate analysis to determine the value of k such that the amortized cost per operation lies between k and k + 1. Hint: For operation i = 1, cost is 1, operation i = 2, cost is 8, operation i = 3, cost is 1, operation i = 4, cost is 16, operation i = 5, 6 ,7, cost is 1, operation i = 8, cost is 32, and so on. Answer: S1= 4.2+4.4+4.8+.......... floor of ($\log_2$n) terms =4(2+2^2+2^3+....... floor of($\log_2$n) terms) <= 4.2.((2^($\log_2$n)-1)/(2-1)) =8(n-1)=8n-8. S2=1+1+.....n - floor of($\log_2$n) terms <= n-$\log_2$n S=S1+S2 <= 8n-8+n-$\log_2$n =9n-8-$\log_2$n So, 8n<S<9n So k=8 Note. log n is taken with base 2, 2^$\log_2$n=n.	4