QUIZ 2: Operations Research - Advanced Level

Network Algorithms and Graph Theory

Course: USEEN3 - Operations Research
Master: Networks-IoT
Academic Year: 2025-2026
Total Questions: 65

Instructions

This quiz focuses on advanced applications and deeper conceptual understanding of graph algorithms.
Some questions require analysis, synthesis, and problem-solving skills.

Assumptions: n = number of vertices, m = number of edges (or arcs)

Section 1: Computational Complexity & Algorithm Analysis (8 questions)

1. Let $T(n) = 10n^2 + 20n^3 + 5$. Which statements are TRUE? *(Multiple answers)*

A) $T(n) = \Theta(n^3)$
B) $T(n) = \Theta(n^2)$
C) $T(n) = O(n^3)$
D) $T(n) = \Omega(n^3)$
E) $T(n) = O(n^2)$

Answer: A, C, D

2. When comparing two algorithms A ($O(n+m)$) and B ($O(n^2 \log n)$) on a sparse tree:

A) Algorithm A is always faster
B) Algorithm B is always faster
C) A is faster since m = n-1 for a tree
D) B is faster for large n

Answer: C

3. For a complete graph K_n, if algorithm A runs in $O(m)$ and algorithm B in $O(n \log n)$:

A) A will always be faster
B) B will always be faster
C) Their performance depends on n
D) B is better since $m = n(n-1)/2$

Answer: D

4. Given a graph with m edges and n vertices where $m >> n^2$:

A) The graph is sparse
B) The graph is dense
C) The graph may have multiple edges
D) The graph must be directed

Answer: C

5. If an algorithm runs in $O(n + m)$ on a sparse graph and $O(n \log n)$ on a tree, what is its behavior on a complete graph?

A) $O(n \log n)$
B) $O(n + m) = O(n^2)$
C) $O(n^2)$
D) Cannot determine without more info

Answer: C

6. Space complexity comparison:

For a graph with n vertices stored as adjacency matrix vs. adjacency list:

A) Matrix: $O(n^2)$, List: $O(n+m)$
B) Matrix: $O(n+m)$, List: $O(n^2)$
C) Both: $O(n+m)$
D) Both: $O(n^2)$

Answer: A

7. Finding all edges incident to a vertex v:

Using adjacency matrix: $O(n)$
Using adjacency list: $O(\text{degree}(v))$

For a complete graph where $\text{degree}(v) = n-1$:

A) List is always better
B) Matrix is always better
C) Both have the same complexity
D) It depends on the value of n

Answer: C

8. Which algorithm would you choose for finding shortest paths in a road network with occasional negative weights (tolls)?

A) Dijkstra
B) Bellman-Ford
C) BFS
D) DFS

Answer: B

Section 2: Graph Structure & Properties (10 questions)

9. In a connected undirected graph with n=100 vertices and m=150 edges:

A) Must be a tree
B) Must have cycles
C) Must be bipartite
D) Could be either acyclic or cyclic

Answer: B

10. A directed acyclic graph (DAG) with n vertices can have at most how many edges?

A) n-1
B) n^2
C) n(n-1)/2
D) 2^n

Answer: C

11. The Handshaking Lemma states that sum of all degrees equals:

A) n
B) m
C) 2m
D) n+m

Answer: C

12. In a directed graph, the sum of all indegrees equals:

A) Sum of all outdegrees
B) Twice the number of edges
C) n(n-1)
D) Cannot determine

Answer: A

13. A graph G has a Hamiltonian cycle if and only if:

A) It has a spanning tree
B) It is connected and every vertex has degree ≥ 2
C) There exists a cycle visiting each vertex exactly once
D) It is complete

Answer: C

14. An Eulerian path visits:

A) Every vertex exactly once
B) Every edge exactly once
C) Every vertex at least twice
D) The entire graph

Answer: B

15. In a complete bipartite graph $K_{m,n}$:

A) Total vertices = m+n
B) Total edges = m·n
C) It is bipartite
D) All of the above

Answer: D

16. Two graphs are isomorphic if:

A) They have the same number of vertices
B) They have the same number of edges
C) They have the same degree sequence and structure
D) They have identical adjacency matrices

Answer: C

17. A tree is a connected graph that:

A) Has n-1 edges
B) Has no cycles
C) Is acyclic
D) All of the above

Answer: D

18. In a directed graph, a strongly connected component (SCC):

A) May contain multiple cycles
B) Every vertex can reach every other vertex in the SCC
C) Is maximal with this property
D) All of the above

Answer: D

Section 3: Graph Search & Traversal (12 questions)

19. When performing BFS from source s, vertices are discovered:

A) In order of their distances from s
B) In arbitrary order

C) In decreasing order of distance
D) Based on lexicographic ordering

Answer: A

20. In DFS, the "back edge" is defined as:

A) An edge to an already visited vertex
B) An edge from descendant to ancestor in DFS tree
C) An edge returning to the source
D) An edge with weight 0

Answer: B

21. DFS can detect cycles by finding:

A) Forward edges
B) Back edges
C) Cross edges
D) Tree edges

Answer: B

22. The time complexity of finding connected components using DFS or BFS is:

A) O(n)
B) O(m)
C) O(n+m)
D) O(n log m)

Answer: C

23. If a graph has k connected components, how many times must we call BFS/DFS?

A) Once
B) k times (at most)
C) n times
D) m times

Answer: B

24. Topological sort using DFS processes vertices:

A) In order of discovery time
B) In reverse order of finishing time
C) In increasing order of indegree
D) Randomly

Answer: B

25. For a DAG with n vertices, how many vertices can have indegree 0?

A) Exactly 1
B) At least 1
C) At most n-1
D) Cannot determine

Answer: B

26. Tarjan's algorithm for finding strongly connected components runs in:

A) O(n)
B) O(m)
C) O(n+m)
D) $O(n^2 \log n)$

Answer: C

27. BFS ensures that when we visit a vertex v:

A) All vertices closer to s are already visited
B) We have found the longest path
C) We are processing its neighbors
D) Its degree has been computed

Answer: A

28. In DFS, if we encounter an edge (u,v) where v is in the current recursion stack:

A) We have found a cycle
B) v is an ancestor of u
C) We should terminate
D) A and B

Answer: D

29. The finish time in DFS is when:

A) We first visit a vertex
B) We completely finish exploring from that vertex
C) We find an edge to an unvisited vertex
D) We return to the source

Answer: B

30. In BFS, the distance array dist[v] represents:

A) Shortest path weight to v
B) Number of edges in shortest path to v
C) Total distance traveled
D) Degree of v

Answer: B

Section 4: Shortest Path Algorithms (15 questions)

31. Dijkstra's algorithm requires:

A) Non-negative edge weights
B) At least one source vertex
C) Connected graph
D) A and B

Answer: D

32. In Dijkstra's algorithm, once a vertex is added to the "finished" set:

A) Its distance may still decrease
B) Its distance is final
C) Its neighbors will be processed
D) It will not be revisited

Answer: B

33. The relaxation operation in Dijkstra updates dist[v] if:

A) $dist[u] + w(u,v) < dist[v]$
B) $dist[u] + w(u,v) > dist[v]$
C) $w(u,v)$ is negative
D) v is unvisited

Answer: A

34. Bellman-Ford algorithm time complexity is:

A) $O(nm)$
B) $O(n+m)$
C) $O(n^2 \log n)$
D) $O(m \log n)$

Answer: A

35. Bellman-Ford detects negative cycles by:

A) Checking if any edge can be relaxed after n-1 iterations
B) Checking if all vertices are reachable
C) Computing all-pairs shortest paths
D) Finding back edges

Answer: A

36. In Bellman-Ford, SP(v, i) represents:

A) Shortest path to v using exactly i edges
B) Shortest path to v using at most i edges
C) Path from v with i hops
D) i-th shortest path to v

Answer: B

37. Floyd-Warshall algorithm computes:

A) Single-source shortest paths
B) All-pairs shortest paths
C) Minimum spanning tree
D) Topological ordering

Answer: B

38. Floyd-Warshall time complexity is:

A) O(n+m)
B) $O(n^2)$
C) $O(n^3)$
D) O(m log n)

Answer: C

39. The relaxation equation dist[v] = min(dist[v], dist[u] + w(u,v)) is used in:

A) Dijkstra
B) Bellman-Ford
C) Floyd-Warshall
D) All of the above

Answer: D

40. Why can't Dijkstra handle negative edge weights?

A) It becomes too slow
B) Greedy choice of closest vertex may not be optimal
C) It requires infinite time
D) It's a matter of convention

Answer: B

41. A priority queue in Dijkstra should support:

A) Insert and extract-min
B) Decrease-key
C) Delete
D) A and B

Answer: D

42. Using a Fibonacci heap, Dijkstra runs in:

A) O((n+m) log n)
B) O(n + m log n)
C) O(n log n + m)
D) O(nm)

Answer: C

43. The Bellman-Ford algorithm can be applied to:

A) Graphs with positive weights only
B) Graphs with negative weights but no negative cycles
C) Directed acyclic graphs
D) B and C

Answer: D

44. In routing protocol RIP, the maximum number of hops is:

A) 10
B) 15
C) 16
D) 32

Answer: B

45. Why is Bellman-Ford used in RIP instead of Dijkstra?

A) It's simpler to implement
B) Routers don't know full topology
C) RIP uses distance vectors
D) All of the above

Answer: D

Section 5: Minimum Spanning Trees (10 questions)

46. Prim's algorithm grows the MST by:

A) Adding the smallest edge overall
B) Adding the smallest edge connecting the tree to a new vertex
C) Processing vertices in order
D) Processing edges by weight

Answer: B

47. Kruskal's algorithm grows the MST by:

A) Always starting from a vertex
B) Selecting edges in increasing weight order without creating cycles
C) Building from multiple components
D) B and C

Answer: D

48. Using a binary heap, Prim's algorithm runs in:

A) $O(n^2)$
B) O((n+m) log n)
C) O(m log m)
D) O(n log n)

Answer: B

49. Kruskal's algorithm requires:

A) A priority queue
B) A union-find data structure
C) A binary tree
D) An adjacency matrix

Answer: B

50. If all edge weights are identical:

A) MST is unique
B) There can be multiple different MSTs
C) MST has maximum edges
D) MST is a complete graph

Answer: B

51. The number of edges in any MST of a connected graph with n vertices is:

A) n
B) n-1
C) n+1
D) 2n-1

Answer: B

52. The cut property states that:

A) The minimum weight edge crossing any cut is in some MST
B) All MSTs have the same weight
C) We can remove any edge from MST
D) A and B

Answer: D

53. Borůvka's algorithm for MST:

A) Is similar to Prim's
B) Is similar to Kruskal's
C) Processes multiple components in parallel
D) Uses a different approach than Prim/Kruskal

Answer: C

54. If we add a new edge to an MST:

A) The graph remains a tree
B) Exactly one cycle is created
C) The MST must be recomputed
D) B and C

Answer: B

55. To find the heaviest edge in the unique path between two vertices in an MST:

A) Use BFS
B) Use DFS
C) Use Dijkstra
D) Use any tree traversal

Answer: D

Section 6: Routing Protocols & Network Applications (10 questions)

56. OSPF (Open Shortest Path First) is:

A) A distance-vector protocol
B) A link-state protocol
C) A path-vector protocol
D) A hybrid protocol

Answer: B

57. RIP (Routing Information Protocol) uses:

A) Dijkstra on full topology
B) Bellman-Ford distributed algorithm
C) Link-state flooding
D) Path vectors

Answer: B

58. In OSPF, each router floods:

A) Complete routing tables
B) Link state advertisements (LSA)
C) Distance vectors
D) Topology changes

Answer: B

59. RIP routers exchange routing tables:

A) When topology changes
B) Periodically (e.g., every 30 seconds)
C) On request
D) A and B

Answer: D

60. The maximum metric (distance) in RIP is 15 hops, which means:

A) RIP works only on local networks
B) RIP limits network diameter to 15 hops
C) Unreachable destinations are marked as 16
D) All of the above

Answer: D

61. BGP (Border Gateway Protocol) operates:

A) Within an AS
B) Between AS
C) At the application layer
D) At the physical layer

Answer: B

62. OSPF Areas are created to:

A) Increase routing table size
B) Reduce routing complexity in large networks
C) Support multicast
D) Handle negative cycles

Answer: B

63. In OSPF, the backbone area is:

A) Area 0
B) Area 1
C) Area 255
D) Any designated area

Answer: A

64. An Autonomous System (AS) is:

A) A group of networks under single administrative control
B) Always owned by one ISP
C) A physical network segment
D) A firewall configuration

Answer: A

65. Comparing OSPF and RIP:

A) OSPF scales better than RIP
B) RIP uses less CPU
C) OSPF converges faster
D) All of the above

Answer: D

Additional Problem-Solving Questions (10 questions)

Problem Set 1: Algorithm Selection

Q1. You have a weighted graph with positive and negative edges (no negative cycles).
Which algorithm(s) would you use and why?

Answer: Bellman-Ford because it handles negative weights. Dijkstra cannot be used directly due to negative edges.

Q2. You have a massive network topology with millions of routers.
Which routing protocol is more suitable: OSPF or RIP? Justify.

Answer: OSPF is better because:

- RIP's 15-hop limit would restrict network size
- OSPF uses link-state (scales better)
- OSPF converges faster with changes
- RIP has higher overhead with periodic updates

Q3. You want to find all connected components in an undirected graph efficiently.

Answer: Use BFS or DFS: O(n+m) time. Call BFS/DFS from each unvisited vertex to mark connected components.

Problem Set 2: Graph Construction & Analysis

Q4. Given a directed graph with n vertices. What's the maximum number of strongly connected components it can have?

Answer: n (each vertex is its own SCC if no edges between them)

Q5. A tree has n vertices. After adding m new edges, how many cycles can be created at minimum?

Answer: m (each new edge creates exactly one fundamental cycle)

Q6. In a complete graph K_n, how many Hamiltonian cycles exist?

Answer: (n-1)!/2 distinct cycles (accounting for direction/starting point symmetry)

Problem Set 3: Complexity Analysis

Q7. Compare the time to find an MST on a sparse graph ($m = O(n)$) vs dense graph ($m = O(n^2)$) using Kruskal's algorithm with union-find.

Graph Type	Sorting Cost	Union-Find Cost	Total
Sparse	$O(n \log n)$	$O(n\ \alpha(n))$	$O(n \log n)$
Dense	$O(n^2 \log n)$	$O(n\ \alpha(n))$	$O(n^2 \log n)$

Answer: Sorting dominates; dense graphs take $O(n^2 \log n)$ vs sparse $O(n \log n)$

Q8. Why does Dijkstra with a Fibonacci heap achieve $O(m + n \log n)$?

Answer:

- Each vertex extracted: $O(n \log n)$
- Each edge processed with one decrease-key: $O(m)$ amortized
- Total: $O(m + n \log n)$

Problem Set 4: Practical Scenarios

Q9. A router needs to forward packets in a network with changing link weights (due to congestion). Should it recompute routes with OSPF after every packet? Why or why not?

Answer: No, recomputing after every packet is too expensive.

- OSPF uses a timer (typically 30 seconds or more)
- Routes are recomputed only when significant topology changes occur
- Trade-off between optimality and computational cost

Q10. In a datacenter network, you need to route traffic with some paths having negative "costs" (incentives).

What algorithm would you use and why?

Answer: Bellman-Ford because:

- It handles negative weights
- With distributed implementation, could work in network routing
- Alternative: Floyd-Warshall for all-pairs planning
- Note: RIP/OSPF wouldn't work directly; would need modifications

End of Quiz 2

Key Topics Covered:

- Advanced complexity analysis with different graph types
- Deep understanding of graph properties
- Subtle differences between DFS/BFS behaviors
- Why different shortest path algorithms are chosen
- Practical routing protocol applications
- MST algorithm selection and efficiency
- Problem-solving and algorithm selection skills