

Name:	
NetID:	

- ***Don't panic!***
- If you brought anything except your writing implements, your double-sided **handwritten** (in the original) 8½" × 11" cheat sheet, and your university ID, please put it away for the duration of the exam. In particular, please turn off and put away *all* medically unnecessary electronic devices.
- **Best answer.** Choose best possible choice if multiple options seems correct to you – for algorithms, faster is always better.
- Please ask for clarification if any question is unclear.
- **This exam lasts 75 minutes.**
- Fill your answers in the Scantron form using a pencil. We also recommend you circle/mark your answer in the exam booklet.
- Do not fill more than one answer on the Scantron form - such answers would not be graded. Also, fill your answer once you are sure of your answer – erasing an answer might make the form unscannable.
- ***Good luck!***

Before doing the exam...

- Fill your name and netid in the back of the Scantron form, and also on the top of this page.
- **Fill in the pattern shown on the right in the Scantron form.**

This encodes which version of the exam you are taking, so that we can grade it.

92	●	Ⓐ	Ⓑ	Ⓒ	Ⓓ	Ⓔ
93	●	Ⓐ	Ⓑ	Ⓒ	Ⓓ	Ⓔ
94	●	Ⓐ	Ⓑ	Ⓒ	Ⓓ	Ⓔ
95	Ⓐ	Ⓑ	Ⓒ	●	Ⓓ	Ⓔ
96	Ⓐ	Ⓑ	●	Ⓓ	Ⓔ	

1. (2 points)

Let G be the complete graph with all the edges having weight either 1 or 2. Consider the problem of computing the cheapest simple cycle that goes through all the vertices of G . This problem can be

- (A) 3/2-approximated, in polynomial time, and one can not do better.
 - (B) 2-approximated, in polynomial time, and one can not do better.
 - (C) solved exactly in polynomial time.
 - (D) solved exactly in $O(n^2)$ time using matchings.
 - (E) solved exactly in $O(n^3)$ time using network flow.
-

2. (2 points)

Given a tree T with n vertices, computing the largest independent set in T can be done in time:

- (A) $O(n^3)$.
 - (B) $O(n^3)$.
 - (C) $O(n)$.
 - (D) $O(n \log n)$.
 - (E) $O(n^4)$.
-

3. (2 points)

Let G be an undirected graph with $n > 4$ vertices and $m > 20n$ edges. Which of the following statements is correct:

- (A) There is a matching in G of size ≥ 10 .
 - (B) There is a matching in G of size $\Omega(m)$.
 - (C) There is a matching in G of size $\geq n/2$.
 - (D) There is a matching in G of size $\Omega(\sqrt{m})$.
 - (E) There is a matching in G of size $\geq 40\sqrt{m/n}$.
-

4. (2 points)

Given a directed graph with n vertices and m edges, and positive integer weights on the edges, deciding if there is a path between two vertices of weight exactly W is

- (A) Can be solved in polynomial time.
- (B) None of the other answers.
- (C) Can be solved in linear time.
- (D) Can be solved in the time it takes do Dijkstra in the graph.
- (E) NPC.

5. (2 points)

Let G be a directed graph with weights on the edges, which might be negative or positive. We have the following:

- (A) All the other answers are correct.
 - (B) Computing if there is a negative cycle in G can be done in polynomial time.
 - (C) Computing the shortest simple cycle in G is **NP-HARD**.
 - (D) Computing the longest simple cycle in G is **NP-HARD**.
-

6. (2 points)

Let X be a set of n distinct numbers in $[0, 1]$. Let x_1, \dots, x_k be k numbers of X . Computing the rank of each of the x_i , for $i = 1, \dots, k$, can be done in time (faster is better):

- (A) $O(nk)$.
 - (B) $O(k)$.
 - (C) $O(n \log k)$.
 - (D) $O(n \log n + k \log n)$.
 - (E) $O(k \log n)$.
-

7. (2 points)

Let G be an instance of network flow with all capacities being odd integer numbers. Then, the value of the maximum flow in G must be an odd number.

- (A) **NPC**.
 - (B) True.
 - (C) False.
-

8. (2 points)

Let L be an instance of linear programming with n variables and m constraints, where all the constraints, except for 5 constraints, are equality constraints (the remaining 5 are inequalities). Solving such an LP can be done in

- (A) $O(n^n)$ time using the simplex algorithm.
- (B) This problem is **NPC**.
- (C) Polynomial time.

9. (2 points)

Consider the problem of deciding if a graph has a set X of k edges, such that each vertex is adjacent to an edge in X . This problem is:

- (A) A variant of 3SAT, and it is **NPC**.
 - (B) A variant of 2SAT, and it is solvable in linear time.
 - (C) A variant of Vertex Cover, and it is **NPC**.
 - (D) solvable in polynomial time.
-

10. (2 points)

Let G be a graph with a min cut of size $k \geq 1$.

- (A) The graph G has at least one spanning tree, and no stronger statement can be made.
 - (B) The graph G does not necessarily have a spanning tree.
 - (C) The graph G has at least k edge disjoint spanning trees, and no stronger statement can be made.
-

11. (2 points)

You are given two sets R, B of n points in \mathbb{R}^d , and consider the problem of computing a hyperplane that separates them (say, it passes through the origin). Let Δ be the diameter of $R \cup B$, and let $\ell = \min_{r \in R, b \in B} \|r - b\|$. Which of the following statements is correct.

- (A) since the problem is equivalent to linear programming, and we do not know if there is a strongly polynomial time algorithm for LP, it follows that this can not be solved in polynomial time.
 - (B) the problem is **NP-HARD**.
 - (C) running time $O(n^{\lceil d/2 \rceil})$ is possible, and no faster algorithm is possible.
 - (D) running time $O(n^{d+1})$ is possible, and no faster algorithm is possible.
 - (E) one can compute such a separating hyperplane in time polynomial in Δ and $1/\ell$, d , and n .
-

12. (2 points)

Let I be an instance of network flow, defined over a graph with n vertices and m edges. Let L be the LP modeling the network-flow. The LP L can be solved in time T . Up to constant factors, we have:

- (A) The time T is (asymptotically) more than it takes to solve network flow, but less than the time it takes to solve a general LP with this number of variables and constraints.
- (B) $T =$ Time it takes to solve a general LP with this number of constraints and variables, and not faster.
- (C) $T =$ Time it takes to solve network flow of this size.

13. (2 points)

Let G be an instance of network flow, with n vertices and m edges (with all capacities being integers). Computing acyclic maximum flow in G is

- (A) Doable in polynomial time.
- (B) NPC.

14. (2 points)

Given n random real numbers picked uniformly and independently from the interval $[0, 1]$, one can decide if there are two equal numbers, with high probability, in time (faster is better):

- (A) $O(1)$.
- (B) $O(n \log n)$.
- (C) $O(n)$.
- (D) $O(n \log \log n)$.

15. (2 points)

You are given an undirected graph G with n vertices and m edges. Assume that the min-cut in this graph is of size $k > 1$. Which of the following statements is correct.

- (A) For any pair of vertices u, v in G , there are at least k edge disjoint paths that connect u to v .
- (B) The max-flow in G is of value k .
- (C) Any two vertices in G are on a simple cycle.

16. (2 points)

Let L be an instance of linear programming with n variables and 7 constraints. Computing the value of the optimal solution for such an LP can be solved in

- (A) This problem is NPC.
- (B) Polynomial time.
- (C) $O(n^n)$ time using the simplex algorithm.

17. (2 points)

Consider a random variable X_i , where $X_0 = n$, and $\mathbb{E}[X_i | X_{i-1}] = \lfloor X_{i-1}/4 \rfloor$, for all $i > 0$. Let U be the first index such that $X_U = 0$. Consider the probability that $\mathbb{P}[U \geq \lceil \log_2 n \rceil]$. This probability is bounded by (the smaller the upper bound, the better):

- (A) $1/n^3$.
- (B) $1/n^{40}$.
- (C) $1/n$.
- (D) $1/2$.
- (E) $1/n^2$.

18. (2 points)

Consider an unweighted graph G with diameter $\Delta > 1$ (i.e., the shortest path between any pair of vertices of G is at most Δ , and there is a pair with this distance). There must be an independent set in G of size at least (bigger is better)

- (A) Δ^2
- (B) Δ
- (C) $\lceil (\Delta + 1)/2 \rceil$.
- (D) $\lfloor n/(1 + \Delta) \rfloor$
- (E) n/Δ^2

19. (2 points)

In the shortest-path with colors problem, you are given a weighted undirected graph G , a start vertex s , an end vertex t , and a list of k colors c_1, \dots, c_k that you need to pick. Every vertex has an associated color. A path $\pi = v_1, v_2, \dots, v_t$ from s to t is *valid* if, for any i , $1 \leq i \leq k$, the color c_i appears in some location ℓ_i (i.e., the color of v_{ℓ_i} is c_i). Furthermore, $\ell_1 < \ell_2 < \dots < \ell_k$. The task is to compute the shortest path (not necessarily simple) of this type.

This problem is

- (A) **NP-HARD**.
- (B) This version is **NP-HARD**, and the version where you just have to pick all the colors (but the order does not matter) is also **NP-HARD**.
- (C) Solvable in $O(n \log n + m)$ time, and no faster algorithm is possible.
- (D) Solvable in $O(k(n \log n + m))$ time.
- (E) Solvable in $O(n^3)$ time, and no faster algorithm is possible.

20. (2 points)

Given a set X of n numbers, and an array T of size n^2 . Let \mathcal{H} be a 2-universal family of hash functions into T . A hash function h is good, if no two elements of X collide when mapped by h into T . We have that

- (A) There is at least one good function in \mathcal{H} .
- (B) When 2-universal family of functions sends its functions, they are not sending their best. They are bringing collisions. They are mixing values. They are bad. And some, I assume, are good functions.
- (C) All the functions in \mathcal{H} are good.
- (D) There is not necessarily any good function in \mathcal{H} because of the birthday paradox.
- (E) At least half the functions in \mathcal{H} are good.

21. (2 points)

Consider a weighted instance of set cover (G, \mathcal{F}) that is special – there are n elements in the ground set, and every element appears in at most t sets of \mathcal{F} , where t is some small integer positive constant. Here, each set in \mathcal{F} has an associated positive cost, and the purpose is to find a minimum cost collection of sets that covers G . Here, you can assume that LP can be solved in polynomial time. Which of the following statements is correct (a better approximation is better):

- (A) One can compute a $O(\log t)$ approximation in polynomial time.
- (B) One can compute a $t/2$ approximation in polynomial time.
- (C) One can compute a $O(\log n)$ approximation in polynomial time.
- (D) One can compute a t approximation in polynomial time.

22. (2 points)

Consider k -CNF formula E over n variables, and with $2n$ clauses. Here, every clause has exactly k literals (which are all distinct variables). Consider the case that $k \geq 4 \lceil \lg n \rceil$. We have that

- (A) E always has a satisfying assignment.
- (B) deciding if there is a satisfying assignment for E is **NPC**.
- (C) E might not have a satisfying assignment.

23. (2 points)

Given two sets R and B of n points in \mathbb{R}^3 , one can decide if R can be separated from B in expected time (faster is better):

- (A) $O(n^3)$.
- (B) None of other answers are correct.
- (C) $O(n)$.
- (D) $O(n^n)$.
- (E) $O(n \log n)$.

24. (2 points)

Let G be an undirected unweighted graph with $n > 1$ vertices and $m > 1$ edges. A *minimal cycle* is a cycle in G such that no proper subset of its vertices form a cycle in G . How fast can one compute a minimal cycle?

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

25. (2 points)

Let X be some optimization problem (e.g., max clique). You are given a reduction *improve* that takes any polynomial time α -approximation algorithm to X , for any $\alpha > 1$, and generates a new polynomial time approximation algorithm, with the quality of approximation being improved to $O(\log \alpha)$. You are given a polynomial time $O(n)$ -approximation algorithm to X . As such, we have the following:

- (A) one can compute an $O(\log^* n)$ -approximation to X in polynomial time.
- (B) X is **NP-HARD** to approximate within a constant factor.
- (C) one can compute an $O(1)$ -approximation to X in polynomial time.
- (D) X is **NP-HARD** to approximate within any factor that is smaller than $\log n$.
- (E) one can compute an $O(\log \log \log n)$ -approximation to X in polynomial time.

26. (2 points)

Let X be a set of n distinct numbers in $[0, 1]$. You are given an oracle that can in constant time answer the following two queries:

- (A) Given parameters $\alpha \leq \beta$, the oracle returns how many elements of X are in the interval $[\alpha, \beta]$.
- (B) Given parameters $\alpha \leq \beta$, the oracle returns a random number in X that lies in the interval $[\alpha, \beta]$ (the random number is chosen uniformly among all such numbers).

Computing the median of X can be done in expected time (faster is better):

- (A) $O(\log^3 n)$.
- (B) $O(1)$.
- (C) $O(\log^2 n)$.
- (D) $O(\log n)$.
- (E) $O(n)$.

27. (2 points)

Given a set X of n positive integer numbers, there is always a hashing scheme that requires $O(1)$ time per operation, and can store X using $O(n)$ space. This statement is

- (A) correct.
- (B) Mostly correct.
- (C) Incorrect.

28. (2 points)

Let G be an integral instance of network flow, with n vertices and m edges. Let s be the source, and let K be the value of the maximum flow in G . Let τ be a parameter. Deciding if there is a maximum flow in G of value K that uses only τ edges coming out of s (i.e., all the other edges from s have zero flow on them) is

- (A) **NPC**.
- (B) Can be done in polynomial time using maximum flow.

29. (2 points)

Given an algorithm that can compute a 4-coloring of a graph (if such a coloring exists) in polynomial time, would imply that

- (A) None of the other answers.
- (B) All the problems in **NP** can be solved in polynomial time.
- (C) All the problems that are **NPC** can be solved in polynomial time.

30. (2 points)

You are given a stream S of n numbers, and space $s = \Omega(n^c)$, where $c \in (0, 1)$ is a constant, one can compute the median of S by reading the stream (smaller is better):

- (A) $O(n^c)$ times.
 - (B) $O(\sqrt{n})$ times.
 - (C) $O(n^{1/c})$ times.
 - (D) $O(\log^2 n)$ times.
 - (E) $O(1/c)$ times.
-

31. (2 points)

Given a graph G with n vertices and m edges, computing a vertex cover in G , of size $2k$, if there is such a cover of size k , can be done in time (faster is better):

- (A) $O(k^{m+n}(n+m))$
 - (B) $O(n^k(n+m))$
 - (C) $O(m)$
 - (D) $O(2^{2k}(n+m))$
 - (E) $O(n)$
-

32. (2 points)

Let Q be a set of n points in the plane. Let q_1, \dots, q_n be a random permutation of the points, and let $\tau_i = \max_{b < c \leq i} \|q_b - q_c\|$ be the diameter of q_1, \dots, q_i , for $i = 2, \dots, n$. (Note, that the pair of points realizing the diameter of Q might not be unique.) Let $\beta_i = \mathbb{P}[\tau_i \neq \tau_{i-1}]$. For $i > 2$, we have that

- (A) $\beta_i = 1 - 2/i$.
 - (B) $\beta_i \leq 2/i$.
 - (C) $\beta_i \geq 1 - 2/i$.
 - (D) $\beta_i = 0$.
 - (E) $\beta_i \leq 1/i$.
-

33. (2 points)

Let G be a DAG over n vertices, and let H be the undirected version of G . Which is of the following statements is correct:

- (A) There is always a (simple) path in G of length $\lfloor \sqrt{n} \rfloor$, or alternatively there is an independent set in H of size $\lfloor \sqrt{n} \rfloor$.
- (B) There is always a (simple) path in H of length $\lfloor \sqrt{n} \rfloor$, or alternatively there is an independent set in H of size $n/2$.

34. (2 points)

Given an unweighted undirected graph G , and parameters k and r , consider the problem of deciding if there is a set S of k vertices in G , such that all vertices in G are in distance at most r from some vertex of S . Which of the following statements is false.

- (A) This problem is **NPC** even if the graph is a tree.
- (B) If there is such a set, then one can compute efficiently a set S' that has the same property, but with distance $2r$.
- (C) This problem can be solved in polynomial time for $r \geq n/10$.
- (D) This problem is **NPC** even for $r = 1$.

35. (2 points)

Consider the problem of given three strings s_1, s_2, s_3 of total length n , computing the longest string T that is contained in all three strings. Here T is contained in s_i , if one can delete characters in s_i to get T , for $i = 1, 2, 3$. This problem can be solved in (faster is better):

- (A) $O(n^4)$.
- (B) $O(n^3)$.
- (C) $O(n \log n)$.
- (D) $O(n)$.
- (E) $O(n^2)$.

36. (2 points)

Consider a given directed graph G with n vertices and m edges. A set of vertices S in G is influential, if for any vertex $v \in V(G)$, there is a vertex $s \in S$, such that there is a path in G from s to v . One can compute in polynomial time (under the assumption that $P \neq NP$) an α -approximation to the smallest influential set, where the value of α is (smaller is better):

- (A) $O(\log \log n)$
- (B) $O(\log n)$
- (C) $O(n)$
- (D) 2
- (E) $O(\log^* n)$

37. (2 points)

Consider a weighted undirected graph G over n vertices, and let s be a vertex in G . Let v_1, v_2, \dots, v_n be a random permutation of the vertices. For any two vertices $x, y \in V(G)$, let $d(x, y)$ denote the length of the shortest path in G between x and y . For $i = 1, \dots, n$, let n_i be the closest vertex to s in G among the vertices of $V_i = \{v_1, \dots, v_i\}$ (i.e., $n_i = \arg \min_{v \in V_i} d(s, v)$). Similarly, let $f_i = \arg \max_{v \in V_i} d(s, v)$ be the furthest neighbor in V_i , for all i . Let $\ell_i = (d(s, n_i) + d(s, f_i))/2$.

Assume that all the pairwise distances in the graph are unique.

- (A) The sequence ℓ_1, \dots, ℓ_n has $\Theta(\log n)$ distinct values with high probability.
- (B) The sequence ℓ_1, \dots, ℓ_n can have only 2 distinct values.
- (C) All of the other answers are correct.
- (D) The sequence ℓ_1, \dots, ℓ_n has $\Theta(\log n)$ distinct values in expectation.
- (E) The sequence ℓ_1, \dots, ℓ_n has $O(\log^2 n)$ distinct values in expectation.

38. (2 points)

Let G be an instance of network flow with all capacities being rational numbers. Then, the Ford-Fulkerson method would always stop and compute the maximum flow in G . This statement is

- (A) False.
- (B) True.

39. (2 points)

You are given an undirected graph G with n vertices and m edges. Furthermore, assume you are given an oracle, such that given an input graph G , it returns you an edge that IS in the min-cut of G (this takes constant time). In this specific case, you can assume the graph is provided via an adjacency matrix. Given such an oracle, one can compute the min-cut in the graph in (faster is better):

- (A) $O(k)$, where k is the size of the min-cut.
- (B) $O(m)$.
- (C) $O(n^3)$.
- (D) $O(n)$.
- (E) $O(n^4)$.

40. (2 points)

Consider a graph G with n vertices and m edges. Let k be some positive integer number. A k multi-way cut, is a partition of the vertices of G into k sets (S_1, \dots, S_k) . An edge is in the cut, if its endpoints are in different sets of this partition.

- (A) There is always a multi-way cut with at least $m/2$ edges, and no stronger statement can be made
- (B) There is always a multi-way cut with at least m/\sqrt{k} edges, and no stronger statement can be made
- (C) There is always a multi-way cut with at least $\sqrt{m/k}$ edges, and no stronger statement can be made
- (D) There is always a multi-way cut with at least m/k edges, and no stronger statement can be made
- (E) There is always a multi-way cut with at least $(1 - 1/k)m$ edges, and no stronger statement can be made.

41. (2 points)

Let X, Y, Z be three sets of distinct numbers such that $X, Y, Z \subseteq \{1, 2, \dots, n\}$. Consider the problem of deciding if for all $z \in Z$, there exists $x \in X$ and $y \in Y$, such that $z = x + y$. This problem can be solved in time (faster is better, naturally):

- (A) $O(n^2 \log n)$.
- (B) $O(n)$.
- (C) $O(n^2 \log^2 n)$.
- (D) $O(n^2)$.
- (E) $O(n \log n)$.

42. (2 points)

Consider an input $X \equiv x_1, \dots, x_n$ of n numbers, and let k be a parameter (which is relatively small compared to n). Let Y be the sequence X after it is being sorted. Assume that every number in X is at most k locations away from its location in Y . The sorted list Y can be computed by a sorting network of depth (smaller is better):

- (A) $O(\log^2 n)$.
- (B) $O(k)$.
- (C) $O(n \log n)$.
- (D) $O(n)$.
- (E) $O(\log^2 k)$.

43. (2 points)

Let L be an LP, and let L^* be its dual. Which of the following scenarios are possible:

- (A) It can be that L is infeasible, and L^* is feasible.
 - (B) All four other cases are possible.
 - (C) L is not feasible and L^* is not feasible.
 - (D) L is feasible and L^* is feasible.
 - (E) It can be that L is feasible, and L^* is not.
-

44. (2 points)

Deciding if a 3SAT formula F with n variables and m clauses is satisfiable can be done in time (faster is better):

- (A) $O((n + m)^2)$
 - (B) $O(n + m)$
 - (C) $O(2^{n-m})$
 - (D) $O(2^n m)$
 - (E) None of the other answers.
-

45. (2 points)

You are given an algorithm `alg` for an **NP** problem X , which is a polynomial time certifier/verifier. Furthermore, assume that any positive instance of X has a certificate of length $\Theta(\log^2 n)$. Then, we have that

- (A) X is **NPC**.
 - (B) If X is **NPC**, then all the problems in **NP** have $O(\log^2 n)$ length certificate.
 - (C) X can be solved in polynomial time.
 - (D) None of the other answers.
-

46. (2 points)

Consider a graph G over n vertices, with an ordering of the vertices v_1, v_2, \dots, v_n , such that v_i has at most k edges to $\{v_1, \dots, v_{i-1}\}$, for all i . We have that:

- (A) All of the other answers are correct.
- (B) G can be colored using $k + 1$ colors.
- (C) G has at most kn edges.
- (D) G has an independent set of size $\geq n/(k + 1)$

47. (2 points)

Let G be an undirected graph with $n > 1$ vertices and $m > 1$ edges. Let M be a matching in G , such that any augmenting path for M has length at most $2k + 1$. Let opt be the maximum matching in G . We have that :

- (A) $|M| \geq (1 - 1/(4k + 1)) |\text{opt}|$.
- (B) $|M| \leq (1 - 1/(k + 1)) |\text{opt}|$.
- (C) $|M| \leq (1 - 1/(4k + 1)) |\text{opt}|$.
- (D) None of the other answers are correct.
- (E) $|M| \geq (1 - 1/(k + 1)) |\text{opt}|$.

48. (2 points)

Consider a randomized algorithm that in the i th iteration, with probability $1/i^2$ has to perform $O(i^2)$ work, and otherwise it performs $O(\log i)$ work. The expected running time of this algorithm, over n iterations, is (smaller is better):

- (A) $O(n^4)$.
- (B) $O(n^3)$.
- (C) $O(n^2)$.
- (D) $O(n \log n)$.
- (E) $O(n \log^2 n)$.

49. (2 points)

Let G be an undirected graph with n vertices and m edges, that does not have an odd cycle. Then one can compute in linear time an independent set in G of size at least (bigger is better)

- (A) $\lceil n/2 \rceil$.
- (B) $\Theta(\log n)$.
- (C) \sqrt{n} .
- (D) None of the other answers are correct.
- (E) n .

50. (2 points)

There are known sorting networks that sort all binary strings correctly except for one string (yeh, wow). You are given a sorting network N , with n inputs and m gates, and parameters $\delta \in (0, 1)$ and $\varepsilon \in (0, 1)$. Verifying that N works correctly on a specific input can be done in $O(n + m)$ time. You want an algorithm that either finds an input for which N is wrong, or alternatively, the number of inputs for which N is wrong is at most $\delta 2^n$ (out of the 2^n possible binary inputs). An algorithm that makes such a decision, and is correct with probability at least $1 - \varepsilon$, runs in (faster is better):

- (A) $O((n + m)\delta 2^m \log \frac{1}{\varepsilon})$.
- (B) $O((n + m)/(\delta\varepsilon))$.
- (C) $O(\log^2((n + m)/\delta\varepsilon))$.
- (D) $O((n + m)\delta 2^n/\varepsilon)$.
- (E) $O((n + m)\frac{1}{\delta} \log \frac{1}{\varepsilon})$.