Final: Tuesday, December 11, 14:00-15:15, 2018 DCL 1310

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 ● Ⓑ Ⓒ Ⓓ Ⓔ

7

1. (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 $c_1, \ldots, c_k$ that you need to pick. Every vertex has an associated color. A path $\pi = v_1, v_2, \ldots, v_t$ 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, $\ell_1 < \ell_2 < \cdots \ell_k$. The task is to compute the shortest path (not necessarily simple) of this type.

This problem is

(A) Solvable in $O(k(n \log n + m))$ time.

(B) Solvable in $O(n^3)$ time, and no faster algorithm is possible.

(C) Solvable in $O(n \log n + m)$ time, and no faster algorithm is possible.

(D) NP-Hard.

(E) 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.

---

2. (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) All the problems in NP can be solved in polynomial time.

(B) None of the other answers.

(C) All the problems that are NPC can be solved in polynomial time.

---

3. (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 has at least k edge disjoint spanning trees, and no stronger statement can be made.

(C) The graph G does not necessarily has a spanning tree.

---

4. (2 points)

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

(A) All four other cases are possible.

(B) L is not feasible and L^* is not feasible.

(C) L is feasible and L^* is feasible.

(D) It can be that L is feasible, and L^* is not.

(E) It can be that L is infeasible, and L^* is feasible.

5. (2 points)
Let G be an instance of network flow, with n vertices and m edges. Computing acyclic maximum flow in G that has non-zero flow going through all the vertices in the graph (excluding the source and the sink) is

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

---

6. (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) Any two vertices in G are on a simple cycle.
(C) The max-flow in G is of value k.

---

7. (2 points)
Given two sets R and B of n points in $\mathbb{R}^3$, one can decide if there is a point of R inside the convex-hull of B. This can be done in expected time (faster is better):

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

---

8. (2 points)
Let X be some optimization problem (e.g., coloring). 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 $1 + 1/(\alpha - 1)$. You are given a polynomial time β-approximation algorithm to X, where $\beta > 1$. As such, we have the following:

(A) X is NP-Hard to approximate within a constant factor.
(B) one can compute an $O(\log \log \log n)$-approximation to X in polynomial time, and no better approximation is possible.
(C) one can compute an 2-approximation to X in polynomial time, and no better approximation is possible.
(D) one can compute an $O(\log^* n)$-approximation to X in polynomial time, and no better approximation is possible.
(E) one can compute an $(1 + \varepsilon)$-approximation to X in polynomial time, for any constant $\varepsilon \in (0, 1)$.

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

---

10. (2 points)
Let L be an instance of linear programming with n variables and m constraints, where all the constraints, except for five, are equality constraints (the remaining five are inequalities). Solving such an LP can be done in

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

---

11. (2 points)
Consider a random variable X_i, where $X_0 = n$, and $\mathbb{E}[X_i \mid X_{i-1}] = \lfloor X_{i-1}/4 \rfloor$, 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) $1/n^2$.
(B) $1/n$.
(C) $1/n^{40}$.
(D) $1/2$.
(E) $1/n^3$.

---

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

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

---

14. (2 points)
Let G be an undirected graph with $n > 1$ vertices and $m > 1$ edges. Which of the following statements is correct:

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

---

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

---

16. (2 points)
Given a tree T with n vertices, computing the largest independent set in T can be done in time:

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

---

17. (2 points)
Let X be a set of n distinct numbers in $[0, 1]$. Let $x_1, \ldots, x_k$ be k numbers of X. Computing the rank of each of the x_i, for $i = 1, \ldots, k$, can be done in time (faster is better):

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

---

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

---

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

---

20. (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 $O(\log n)$. Then, we have that

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

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

---

22. (2 points)
Let P be a set of points in the plane, such that all pairwise distances are unique, and $n = |P|$. Let $p_1, \ldots, p_n$ be a random permutation of the points, and let $\Delta_i = \max_{b<c\leq i} \|p_b - p_c\|$ be the diameter of $p_1, \ldots, p_i$, for $i = 2, \ldots n$. Let $\alpha_i = \mathbb{P}[\Delta_i \neq \Delta_{i-1}]$. For $i > 2$, we have that

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

---

23. (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.

---

24. (2 points)
You are given an undirected graph (with multiplicities on the edges) 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 not in some min-cut of G (this takes constant time), if such an edge exists. Given such an oracle, one can compute the min-cut in the graph in time (faster is better):

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

25. (2 points)
Sorting n random numbers picked uniformly from $[0,1]$ can be done in time (faster is better):

(A) why knows?
(B) $O(n)$.
(C) $O(n \log\log n)$.
(D) $O(n \log n)$.

---

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

---

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

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

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

---

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

---

31. (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 2SAT, and it is solvable in linear time.
(B) solvable in polynomial time.
(C) A variant of Vertex Cover, and it is NPC.
(D) A variant of 3SAT, and it is NPC.

---

32. (2 points)

Consider an input $X \equiv x_1, \ldots, x_n$ 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) $O(n)$.
(B) $O(\log^2 n)$.
(C) $O(n \log n)$.
(D) $O(k)$.
(E) $O(\log^2 k)$.

---

33. (2 points)

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

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

---

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

35. (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 $n/2$.

---

36. (2 points)
Deciding if a 2SAT formula F with n variables and m clauses, is satisfiable can be done in (faster is better):

(A) $O(2^{n+m})$ time.

(B) None of the other answers.

(C) $O(n+m)$ time.

(D) $O((n+m)^2)$ time.

---

37. (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) If there is such a set, then one can compute efficiently a set S' that has the same property, but with distance $2r$.

(B) This problem is NPC even if the graph is a tree.

(C) This problem is NPC even for $r = 1$.

(D) This problem can be solved in polynomial time for $r \geq n/10$.

---

38. (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 the longest simple cycle in G is NP-Hard.

(C) Computing if there is a negative cycle in G can be done in polynomial time.

(D) Computing the shortest simple cycle in G is NP-Hard.

39. (2 points)
Consider k-CNF formula E over n variables, and with $n^{O(1)}$ clauses. Here, every clause has exactly k literals (which are all distinct variables). Consider the case that $k = 4$. We have that

(A) E always has a satisfying assignment.
(B) deciding if there is a satisfying assignment for E is NPC.

---

40. (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 $v \in \mathsf{V}(\mathsf{G})$, 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 $\mathrm{P} \neq \mathrm{NP}$) an α-approximation to the smallest influential set, where the value of α is (smaller is better):

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

---

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

---

42. (2 points)
Consider a graph G over n vertices such that there is an ordering of the vertices $v_1, v_2, \ldots, v_n$, such that v_i has at most k edges to $\{v_1, \ldots, v_{i-1}\}$, for all i. We have that:

(A) G has at most $kn/2$ edges.
(B) G can be colored using $k/2$ colors.
(C) All of the other answers are incorrect.
(D) G has an independent set of size $\geq 2n/k$
(E) All of the other answers are correct.

43. (2 points)
Consider a randomized algorithm that in the ith 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) $O(n^2)$.
(B) $O(n \log n)$.
(C) $O(n^4)$.
(D) $O(n \log^2 n)$.
(E) $O(n^3)$.

---

44. (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 $(S_1, \ldots, S_k)$. 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 $(1 - 1/k)m$ 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 $m/2$ 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 $\sqrt{m/k}$ edges, and no stronger statement can be made

---

45. (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 (not necessarily simple) between two vertices of weight exactly n^7 is

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

46. (2 points)
Let G be an instance of network flow with all capacities being integer numbers. Then, the maximum flow must assign integral flow to all the edges of G. This statement is

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

---

47. (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) 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.
(B) All the functions in $\mathcal{H}$ are good.
(C) There is at least one good function in $\mathcal{H}$.
(D) At least half the functions in $\mathcal{H}$ are good.
(E) There is not necessarily any good function in $\mathcal{H}$ because of the birthday paradox.

---

48. (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) one can compute such a separating hyperplane in time polynomial in Δ and $1/\ell$, d, and n.
(B) the problem is NP-Hard.
(C) running time $O(n^{d+1})$ is possible, and no faster algorithm is possible.
(D) 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.
(E) running time $O(n^{\lfloor d/2 \rfloor})$ is possible, and no faster algorithm is possible.

49. (2 points)
Let A, B be two sequences of n bits each. Let $A+i$ be the sequence resulting from having a run of i zeroes, followed by the sequence A, and then followed by $n-i$ zeroes (i.e., the sequence $A+i$ is of length $2n$). For two sequence $X = x_1, \ldots, x_n$ and $Y = y_1, \ldots, y_n$, let $\langle X, Y \rangle = \sum_i x_i y_i$ be their dot-product. Computing i and j such that $\langle A+i, B+j \rangle$ ix maximal can be done in time (faster is better):

(A) $O(n^2)$.
(B) $O(n \log n)$.
(C) $O(n)$.
(D) $O(n^{3/2})$.
(E) $O(n^2 \log n)$.

---

50. (2 points)
Consider a weighted undirected graph G over n vertices, and let s be a vertex in G. Let $v_1, v_2, \ldots, v_n$ be a random permutation of the vertices. For any two vertices $x, y \in \mathsf{V}(\mathsf{G})$, let $d(x, y)$ denote the length of the shortest path in G between x and y. For $i = 1, \ldots, n$, let n_i be the closest vertex to s in G among the vertices of $V_i = \{v_1, \ldots, v_i\}$ (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 $\ell_i = \bigl(d(s, n_i) + d(s, f_i)\bigr)/2$.
Assume that all the pairwise distances in the graph are unique.

(A) The sequence $\ell_1, \ldots, \ell_n$ can have only 2 distinct values.
(B) All of the other answers are correct.
(C) The sequence $\ell_1, \ldots, \ell_n$ has $\Theta(\log n)$ distinct values in expectation.
(D) The sequence $\ell_1, \ldots, \ell_n$ has $\Theta(\log n)$ distinct values with high probability.
(E) The sequence $\ell_1, \ldots, \ell_n$ has $O(\log^2 n)$ distinct values in expectation.