CS60001 Advances in Algorithms, Autumn 2008– 09

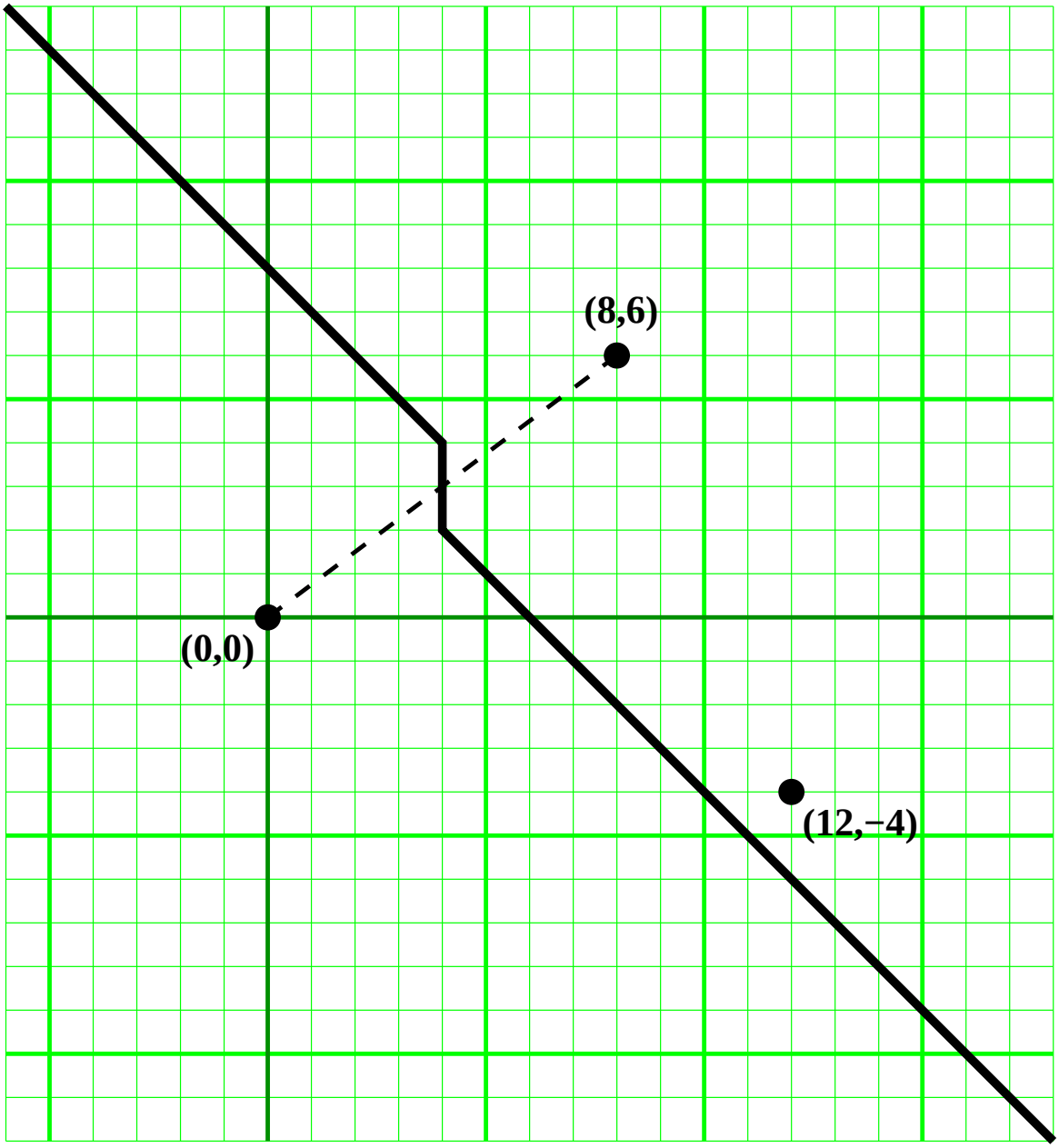
End-Semester Test

Maximum marks: 60 November 22, 2008 Total time: 3 hours

Roll no: _____ **Name:** _____

- *Write your answers in the question paper itself. Be brief and precise. Answer all questions.*
- *Avoid untidiness in the answer script.*

1. The *Manhattan distance* between two points $P_1 = (x_1, y_1)$ and $P_2 = (x_2, y_2)$ in the plane is defined as $d_\infty(P_1, P_2) = \max(|x_1 - x_2|, |y_1 - y_2|)$. Your task is to compute the Voronoi diagram of the three points $P_1 = (0, 0)$, $P_2 = (8, 6)$ and $P_3 = (12, -4)$ with respect to the Manhattan distance. The locus of the points P equidistant from P_1 and P_2 (that is, $d_\infty(P, P_1) = d_\infty(P, P_2)$) is provided. Draw the similar loci for the two other pairs (P_2, P_3) and (P_1, P_3). Then, mark the Voronoi cells of the three points P_1, P_2 and P_3. **(3+3+4)**



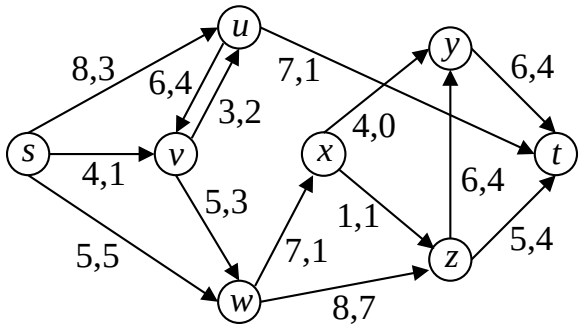
2. An *interval* $[a, b]$ refers to the set of real numbers x satisfying $a \leqslant x \leqslant b$. Two intervals I_1 and I_2 are called overlapping if $I_1 \cap I_2 \neq \emptyset$.

You are given n intervals $I_1 = [a_1, b_1]$, $I_2 = [a_2, b_2]$, $\ldots$, $I_n = [a_n, b_n]$, each specified by its two endpoints. Your task is to find all the pairs (I_i, I_j) of overlapping intervals. Describe an $O(n \log n + h)$-time algorithm to solve this problem, where h is the number of overlapping pairs. You must supply an argument corroborating that your program achieves this running time. You may assume that the endpoints of the input intervals are in general position (no repetitions). **(6+4)**

3. Let $G = (V, E)$ be a simple directed graph with each edge carrying a positive cost. Suppose that each vertex in G also carries a positive cost. You may visualize G as a computer network in which the cost associated with the edge (v, w) stands for the cost of transferring a packet from v to w. On the other hand, the cost associated with a vertex v is the cost of relaying a packet at v. Relaying costs are not applicable at the source and the destination. To sum up, the total cost of the path $v_1, v_2, \ldots, v_k$ is $c(v_1, v_2) + c(v_2) + c(v_2, v_3) + c(v_3) + \cdots + c(v_{k-1}) + c(v_{k-1}, v_k)$.

You are given a distinguished vertex $u \in V(G)$ (the source). Your task is to compute the cheapest paths from u to all vertices in G. Modify Dijkstra's SSSP algorithm to solve this problem. (Mention only the steps you have modified. Do not write the entire algorithm.) Your modification should have the same running time as Dijkstra's algorithm (supply an argument). Also prove the correctness of your modification. **(6+2+2)**

4. Consider the network flow shown in the following figure. Here, s is the source, and t is the sink. The capacity $c(e)$ and the current flow amount $f(e)$ are shown against the edge e as $c(e), f(e)$.



(a) Draw the residual graph for this flow. **(5)**

(b) Mention an s, t path in the residual graph. **(1)**

(c) Redraw the network after augmenting the flow along the path of Part (b). **(4)**

5. Two computational problems P_1 and P_2 are called *polynomial-time equivalent* if there exist polynomial-time reductions $P_1 \leqslant P_2$ and $P_2 \leqslant P_1$. Prove or disprove: Every two NP-Complete problems are polynomial-time equivalent. **(5)**

6. A Boolean formula is said to be in the *disjunctive normal form* (or *DNF* or the *sum-of-products form*) if it is the disjunction (OR) of conjunctions (AND) of literals. For example, $(x_1 \wedge \bar{x}_3) \vee (\bar{x}_2 \wedge \bar{x}_3 \wedge x_4) \vee (\bar{x}_2)$ is in the DNF. By DNF-SAT, we refer to the computational problem of deciding whether a Boolean formula in the DNF is satisfiable. Prove that DNF-SAT $\in \mathrm{P}$. **(5)**

7. Two (simple undirected) graphs $G_1 = (V_1, E_1)$ and $G_2 = (V_2, E_2)$ are said to be *isomorphic* if there exists a bijection $f : V_1 \to V_2$ such that $(u, v) \in E_1$ if and only if $(f(u), f(v)) \in E_2$.

A *subgraph* of a graph $G = (V, E)$ is a graph $H = (V', E')$ with $V' \subseteq V$ and with $(u, v) \in E'$ whenever $(u, v) \in E$ (for $u, v \in V'$). In other words, H is a subgraph of G if its vertex set is a subset of the vertex set of G and if all edges of G with both vertices in $V(H)$ are also edges of H.

Let SUBGRAPH-ISOMORPHISM denote the computational problem to decide, for the input of two graphs G and G', whether G contains a subgraph isomorphic to G'. Prove that SUBGRAPH-ISOMORPHISM is NP-Complete. **(10)**

ROUGH WORK