Algorithms – CS-27200/37000 Homework – February 2, 2004

Instructor: László Babai Ry-164 e-mail: laci@cs.uchicago.edu

ADVICE. Take advantage of the TA's office hours Monday, Tuesday and Thursday 5–6pm in the Theory lounge (Ry–162).

DATES TO REMEMBER. Wed Feb 4: Midterm 1; Mon Feb 23: Quiz 2.

This homework set is **due Friday, February 6** (after the Midterm). The concept of "loop invariants" will NOT be on the midterm.

11.1 (U,G) Let P and Q be statements and S a set of instructions. Consider the loop "**while** P **do** S." Recall that Q is a **loop-invariant** for this loop if for all configurations X (all possible settings of the variables) it is true that

> if $P\&Q$ holds for the configuration X then Q also holds for the configuration $S(X)$,

where $S(X)$ is the configuration obtained from X by executing S.

Note that the highlighted statement has to hold even for infeasible configurations X (i. e., for settings of the variables that could not occur in the course of the execution of the algorithm). The situation has some similarity with chess puzzles: when showing that a certain configuration leads to checkmate in two moves, you do not investigate whether or not the given configuration could arise in an actual game.

Dijkstra's algorithm consists of iterations of a single "**while**" loop. Consider the following two statements:

Q_1 : $(\forall u, v \in V)$(if u is black and v is not black then $c(u) \leq c(v)$).

Q_2 : $(\forall v \in V)(c(v)$ is the minimum cost among all $s \to \ldots \to v$ paths that pass through black vertices only).

(U,G) Prove that Q_1 is a loop-invariant. Prove that $Q_1\&Q_2$ is a loop-invariant. (Do not hand in. Zero points.)

(G, 7 points) Prove that Q_2 alone is *not* a loop-invariant. *Explanation.* You need to construct a weighted directed graph with nonnegative weights, a source, and an assignments of all the variables (parent pointers, status colors, current cost values) such that Q_2 holds for your configuration, but Q_2 will no longer hold after executing Dijkstra's **while** loop. Your graph should have very few vertices (4 vertices suffice).

11.2 (U,G) (3+3+3 points) For each statement, decide whether or not it is a loop-invariant for BFS: (a) "Vertex #2 is black." (b) "Vertex #2 is white." (c) "Vertex #2 cannot change from black to white." Reason your answers!