Algorithms – CS-27200/37000 Homework – January 17, 2005

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

IMPORTANT. If you have not done so yet, please send e-mail to the instructor with your name, major, year, type of credit sought (letter grade, P/F, etc.), list of proof-oriented math courses previously taken; include whether or not you took CMSC-27100 (Discrete Math). In the subject write 27200 info or 37000 info, as approrpiate.

HOMEWORK. Please **print your name on each sheet.** Print "U" next to your name if you seek 27200 credit and "G" if you seek 37000 credit. Undergraduates receive the stated number of points as *bonus points* for "G only" problems. – Please try to make your solutions readable. Unless expressly stated otherwise, all solutions are due at the **beginning of the next class.**

Homework is collected in three separate piles (U, G, "G only"). Please put your solutions to "G only" problems on that pile, and your solutions to other problems on the "U" or "G" pile according to the credit you seek.

5.1 (U,G) (8 points) *(k-way merging.)* Give an $O(n \log k)$-time algorithm to merge k sorted lists into one sorted list, where n is the total number of elements in all the input lists. *Hint.* Use a heap for k-way merging.

5.2 (U, G)

1. (5 points) Write pseudocode to turn the edge-list representation of the digraph $G = (V, E)$ into an adjacency list representation in linear time. ("Linear time" means $O(|V| + |E|)$ steps. Copying an integer between 1 and $|V|$ counts as one step.)
2. (5 points) Write pseudocode to turn an adjacency list representation of the digraph $G = (V, E)$ into a monotone increasing adjacency list representation in linear time, i. e., the out-neighbors of each vertex must be listed in increasing order.
3. (5 points) Write pseudocode to turn an adjacency list representation of the digraph $G = (V, E)$ into an adjacency list representation of the reverse digraph (we reverse each arrow) in linear time.
4. (G only, 5 points) Describe an algorithm to decide, in linear time, whether or not a digraph is undirected (the reverse of every edge is an edge). The digraph is given in adjacency list representation. Your solution should be very simple.