Computer Science 220 S2 (2014)

Assignment 3 (Hashing and Graph Representation)

Due date: Oct 3th, 2014

Problem Statement

Two programs are required to understand basic hashing techniques. We will use open addressing collision resolution with rightward (increase) probing. For a digraph G, let DS be the concatenation of integers (as strings) from a decreasing sorted degree sequence of G.

Program 1: OALP hashing with H_1
Program 2: OADH hashing with H_1 and $\Delta = H_2$

H_1: do middle-squaring on DS.
1) Let d equal the integer obtained from the three middle digits of DS.
(append 0's if length of DS $<$ 3 or even)
2) Hash value is the integer denoted by the three middle digits of d^2.

H_2: do truncation on DS.
1) The initial hash value is the integer denoted by the first three digits of DS plus one.
(prepend 0's if length of DS $<$ 3)
2) For subsequential probes decrease this hash value by one until it is one.

Input will be a sequence of adjacency lists of digraphs (at most 1000) and we want to hash each to an integer using these two hash functions H_1 and H_2. Input will follow the adjacency list format as described in the CS220 textbook. Here, the first line for eaach diagraph is an integer n indicating the order of the digraph. This is followed by n white space separated lists of (out-) adjacencies for nodes labeled 0 to $n - 1$. The last digraph of the sequence will be a digraph of order 0 and this is not to be processed (e.g. not added to hash table).

For both programs output a binary string of length 1000, which is your fixed hash table size. The bit at position i denotes that some digraph was (1/true) or was not (0/false) hashed to value i. Output each bit on a single line using characters '0' and '1' and terminated with a newline '\n'. Thus, your program's output will be fixed at 1000 lines (2000 bytes).

After experimenting with your programs (on several data sets), can you say which program performs better in practice in hashing random sequence of digraphs? Consider cases where there are few (<100) graphs and many (>800) graphs, with respect to your hash table size.

Submission and Due Date

Submit your source code to the automated marker `www.cs.auckland.ac.nz/automated-marker`. Name them like `OALP.java` or like `OADH.java`, where the filename extension `java` may be replaced with extensions `cpp`, `py` or `cs`, for C++, Python, C#/mono compilers. There will be two test cases provided for each problem.

The deadline is 8:30pm (automarker time) on the 3rd of October. The marks for this assignment is worth 8% of your course grade. Some partial credit may be given if you do not achieve "Correct" (green) on the automarker. Note we use plagiarism detection software so do not share any source code with your fellow students. The due date is firm in that there are *no* penalty options available for late submissions (except zero marks).