Comparing linked lists and skip lists.

May 18, 2023

1 Project description

You are going to implement and test a C/C++ program that compares a linked list and a skip list. You are going to implement:

- A linked list. It should have at least the following functionalities: insert a new element, print all elements, remove an element. If using C++ (highly encouraged), data should be private, and accessible from the main program only through calling member functions.
- A skip list. It should have similar functionalities to the ones of the skip list.

You should benchmark and compare the performance of the operations (insert, delete, search) under comparable conditions for the two data structures. You have freedom to the design of the program **Discuss your design choices in the report you submit.**

The following is **a suggestion** for benchmarking experiments:

- Begin by adding a large number (at least 10000) of elements in a linked list. Each element is a pair $(key, value)$, where the key is a string (think of the name of a person in a phone book), and the value is an integer (the phone number). Similarly for the skip list.
- The linked list and the skip list should both be maintained sorted in the lexicographic order of names.
- Report the times for creating the two lists by insertion.
- Create a vector of the names. Each element should contain, in addition to the name, a bit, set to 1.
- Perform the same number of successful searches in the skip list and the linked list. You may choose a random name in the vector and search for the name in both the linked list and the skip list.
- Report the times for successful searches.

- Finally, delete the same number of elements from the two data structures as follows: choose a random element of the vector. If the bit is 1 then delete that name from both the linked list and the skip list, setting the bit to 0. Otherwise choose a new random element.
- Report the times for the deletions.

2 What to turn in

You will turn in

1. *a short written report* containing:
   - A description of the significant choices/issues in the design of your code.
   - Listing of your experiments, showcasing the capabilities of the program.
2. The sourcecode of your program.

You may turn in the document via moodle.unibuc.ro (Class moodle).

3 Coding standards

A percentage of your grade will be based on the quality of your code, so pay attention to it. Discuss changes (if any) you made to programs presented in class. Take extra care in documenting the code you are implementing on your own. Properly modularize the code (for instance implement separate functions for significant parts of the program).

4 Deadline

June 5, 2023, noon local time. This is a strict deadline. No credit will be given for homework turned in late.