Programming 2 Assignment

Gerald Hu, Aaron Skouby

CSCE 221-200

March 11, 2016

Introduction

The purpose of this assignment was to further understanding of data structures and implementations that were discussed in class. Specifically, the project explored implementing a map ADT using a binary tree. The performance of the operations of the map were analayzed for both a normal binary tree implementation and an AVL binary tree implementation.

Implementation Details

Theoretical Analysis

Experimental Setup

Timing tests were conducted using the provided timing.cpp, compiled with the provided makefile's commands. Compilation was done on the "linux.cse.tamu.edu" server, with G++ version 4.7.3 (SUSE Linux) (found via g++ –version). Compilation was set to the C++11 standard, with the -G flag enabled and O2 optimization level, warnings set to -Wall -Werror (all warnings treated as compilation errors), and dependencies flagged with -MMD (auto-generate dependencies).
Tests were run on the "compute.cse.tamu.edu" server, which runs Arch Linux x86_64 version 8.12 (found via arch –version and lsb_release -a). This server has 99026668 total kilobytes of RAM (found via free). It uses Intel Core i7-3970X CPUs (2 sockets, 8 cores per socket, 2 threads per core), with a clock speed of 2000 mHz (found via lscpu). Each core has a Xeon E5-2650 processor (found via lshw –short).
Timing functions output timings for input sizes that were powers of 2, starting from 2 itself, and ending at a maximum size specified by the user. Each step of the timing was repeated 10 times, and the average of each result taken. Linear

height n inserts went up to a maximum input size of 32768; logarithmic height n inserts went up to a maximum input size of 4194304, and random n inserts went up to a maximum input size of 1048576.

Results and Discussion

Conclusion