VE281 Lab2 Report

Haoyun Zhou

November 5, 2022

1 Introduction

In this report, the performance of HashTable implemented in the lab, unordered_map in STL, and list in STL are compared. Same random data is generated for each data structure as the following way. Denote the number of operations by n. Let keys be integers and range from 1 to m. Each operation can be insertion, deletion, and accession in random.

2 Results

When $n = 1000000, m = 1000$, HashTable, unordered_map, and list spends 0.127477s, 0.0958371s, and 4.73161s respectively.

When $n = 100000, m = 1000$, HashTable, unordered_map, and list spends 0.0128269s, 0.0098627s, and 0.470215s respectively.

When $n = 100000, m = 10000$, HashTable, unordered_map, and list spends 0.0152615s, 0.0107248s, and 4.50088s respectively.

3 Discussion

In all cases above, unordered_map spends the shortest time, while list spends the longest time. Time spent by HashTable is close to that of unordered_map.

When $n = 100000$ is fixed and m changes, time consumed by HashTable and unordered_map is in the same order of magnitude, while time consumed by list increases by an order of magnitude as m increases by an order of magnitude. It is because the time complexity of a single operation for HashTable and unordered_map is $O(1)$, while that for list is $O(m)$.

When $m = 1000$ is fixed and n changes from 1000 to 10000, the time consumed by all data structures increase an order of magnitude.

In conclusion, the time complexity of HashTable and unordered_map is $O(n)$, while that for list is $O(nm)$.