Testing Writeup

Assignment #4

CMPT-225

Sameer Hossain – 301357905

**Hardware:** Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz

L1 cache: 384 KB

L2 cache: 1.5 MB

L3 cache: 9.0 MB

**Experiment #1:** Simple Demonstration of Cache Effects

For this experiment, I used an array of size 1 000 000 and modified its elements to compare the CPU time used to between accessing each element of the array and accessing every 16th element and deleted the array after reaching the end.

I used 2 arrays to account for the cache being filled up by the first array access.

Loop 1, where the stride of access is 1, took 2ms and 0.000002ms to access each element.

Loop 2, where the stride is 16, took 1ms and 0.000016ms to access each element.

Text

Description automatically generatedAs we can see the total access time of each loop are quite similar with loop 2 requiring more time to access each element but overall is slightly faster than loop as it accesses less items.

**Experiment #2:** Access times as size of array increases

In this experiment the access times of arrays of multiple sizes are compared to estimate the size of the cache.



From the data, there is a small spike in access time when the size of the array increases from 512KB to 1MB which indicates L1 cache misses and L2 hits. From 1MB to 2MB there is another spike which indicates L3 being accessed.

**Experiment #3:** Traverse times of array, linked list & unrolled linked list

This experiment compares the time taken to traverse a linked list, an unrolled linked list and an array all of which include 1 000 000 elements.

The author’s linked list implementation was used and for the unrolled linked list I modified Sirama Charan’s implementation (Charan, 2020), which was published on codespeedy.com, to suite my test. (Further reference is included in UList.h)

The *insert(int \*a)* function takes an array a and deep copies it to the array inside the node. The node is then inserted to the tail of the list. To calculate the access time of the list, it traverses the nodes incrementing every element of the arrays of each node. For fairness I inserted 1000 nodes each containing an array of 1000 elements totaling to 1 000 000 elements.

Text

Description automatically generatedThe linked list takes the longest time for traversal and unrolled linked list is the fastest.