Kabeen Kim

Professor Byunghyun Jang

Csci 223 Computer Org. & Assembly Language Section 1 2024-2025 FALL

9 September, 2024

Performance Comparison of Data Copying in C++

: Using Static and Dynamic Memory Allocation

I compared the performance of two versions of data copying using both static and dynamic memory allocation in C++. Version 1 is a column-wise access, and Version 2 is a row-wise access. Version 1 is slower than Version 2 for static and dynamic memory allocation due to cache locality. To accurately compare the execution times of the two versions, they are as follows.

1. Static Memory Allocation

Version 1: 2.56062 milliseconds

Version 2: 0.93954 milliseconds

Version 2 is faster by 1.62108 milliseconds.

2. Dynamic Memory Allocation

Version 1: 2.47879 milliseconds

Version 2: 1.03875 milliseconds

Version 2 is faster by 1.44004 milliseconds.

Version 1 is inefficient code according to spatial locality. Because the elements of arrays are stored contiguously in memory, but the access pattern is not sequential. In contrast, Version 2 benefits from the spatial locality, which leads to fewer cache misses and faster execution.

In addition, when memory is allocated statically, it is assigned to the stack using compilation and is released when functions exit. In comparison, when memory is allocated dynamically, it is transferred to the heap. The heap allows the programmer to allocate memory at runtime in the size needed. I didn’t find any specific time difference using static and dynamic memory allocation to distinguish. Whenever I compiled the file again, the execution time always changed and I couldn’t figure out which one was faster in static and dynamic memory allocation. It seems that further study will be required to fully understand this part.