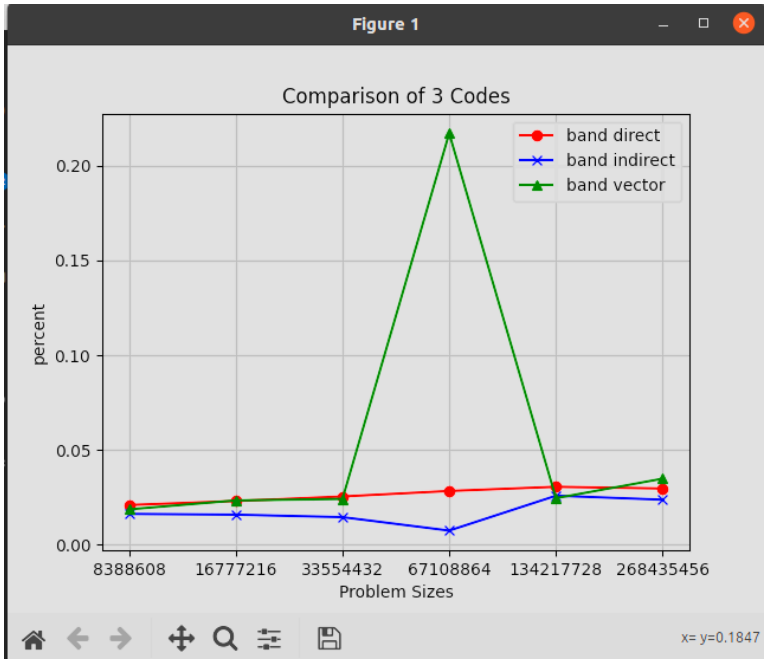
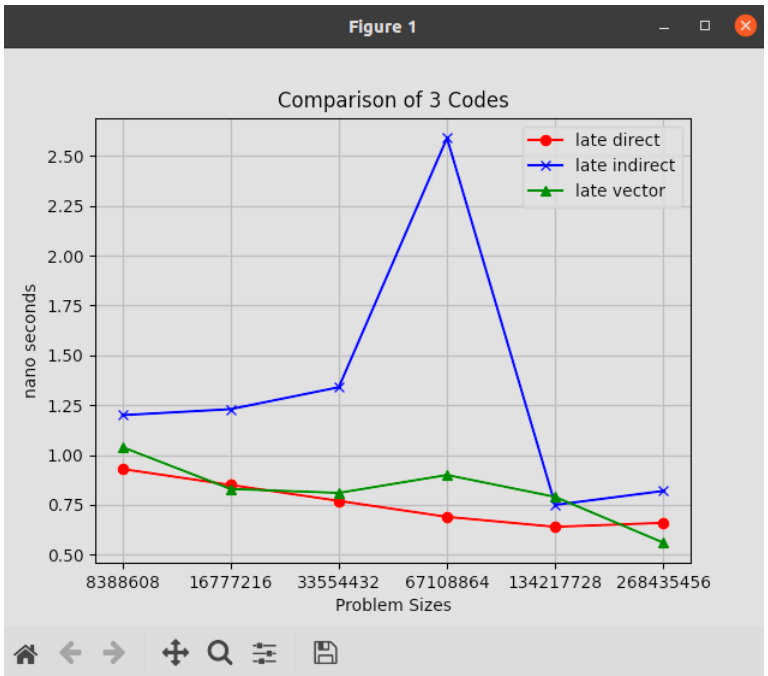
Alec Nagal
CSC 656.01
April 2, 2024

Coding Project #2

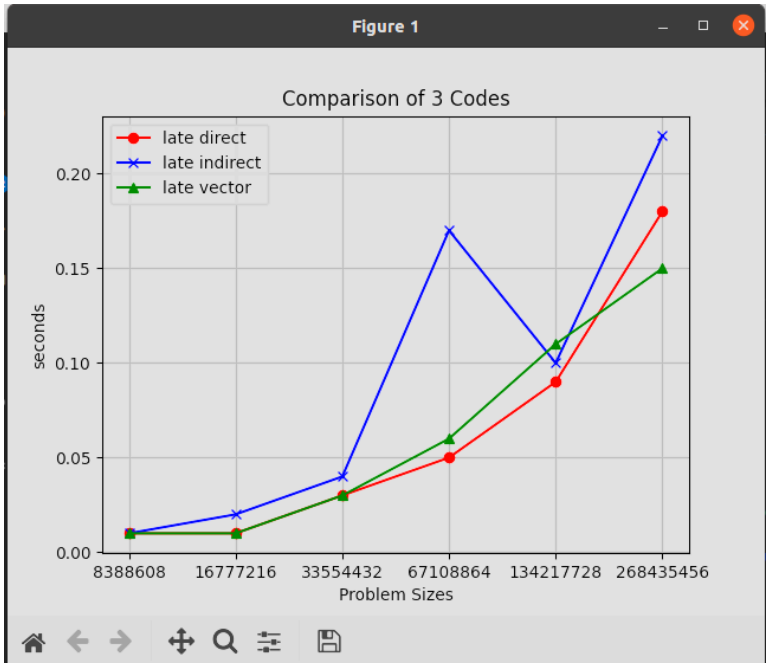
Bandwidth chart



Latency



MFLOP



1. Floating-point operations, especially division and square root, are typically more expensive in terms of CPU cycles than integer operations due to their complexity. Memory operations can also be expensive due to the latency involved in accessing data from RAM. If one of the codes is performing a larger number of floating-point operations or has a higher rate of cache misses, it would considered to be doing more expensive operations.

2. The method with the best computational rate can be determined by examining the output of the benchmark tests. The one with the highest MFLOP/s value is performing the most floating-point operations per second. This might be due to better optimization, such as loop unrolling or because it has less overhead apart from the floating-point operations.

3. Between vector sum and indirect sum, the one with a higher level of memory bandwidth utilization is likely accessing memory more efficiently. This could be due to better contiguous memory access patterns, which enable it to benefit from spatial locality and prefetching. If the access pattern is more random, it could lead to lower bandwidth utilization due to cache misses and delayed memory access.

4. The method that shows lower levels of memory latency would be the one with more efficient memory access patterns. This could be because it accesses memory in a way that's more predictable for the hardware prefetcher, or it could be that it benefits more from temporal locality, reusing data that's already been brought into the cache.

Without specific runtime data, these answers must remain general. To provide detailed responses, we'd need to examine the benchmark results for each method or look at the code to understand how it's accessing memory and what types of operations it's performing.