Name: Sujal Singh
UBITname: sujalsin

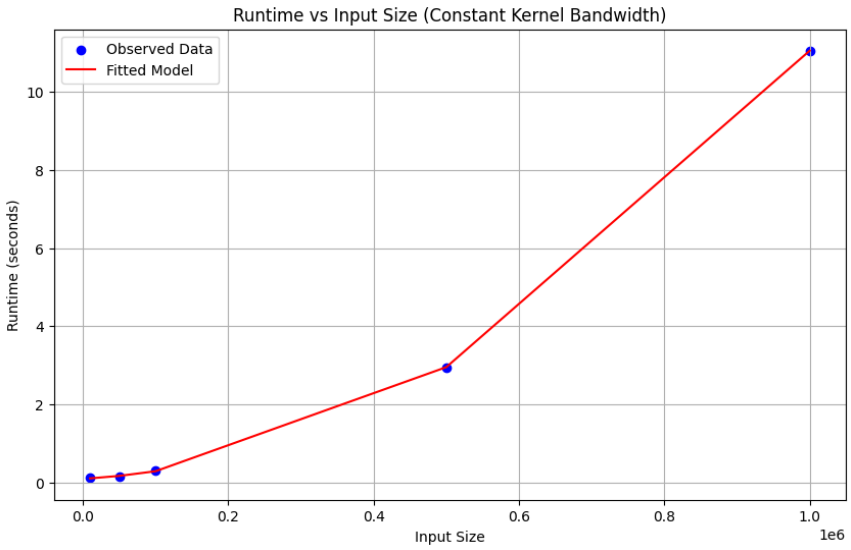
Test Environment Details:

- System: Dell Inc. PowerEdge R440
- Operating System: Ubuntu 22.04.3 LTS (Jammy Jellyfish)
    - Version ID: 22.04
    - Kernel: Linux 5.15.0-84-generic
    - Architecture: x86-64
- GPU: A100
- CPU: AVX512 Supported
- Memory (RAM): 160000 MB
- Compiler: nvcc: NVIDIA (R) Cuda compiler driver (cuda_11.8.r11.8/compiler.31833905_0)
- Compiler: g++ (Gentoo 10.4.0 p5) 10.4.0
- System Information:
    - Hostname: vortex-future.cbls.ccr.buffalo.edu
    - Machine ID: 54febf5842d740da914a81cb26b1fd18
    - Boot ID: a20b9d0a15d046a9b1b6ffd533f28036
    - Chassis: Server

On GPU

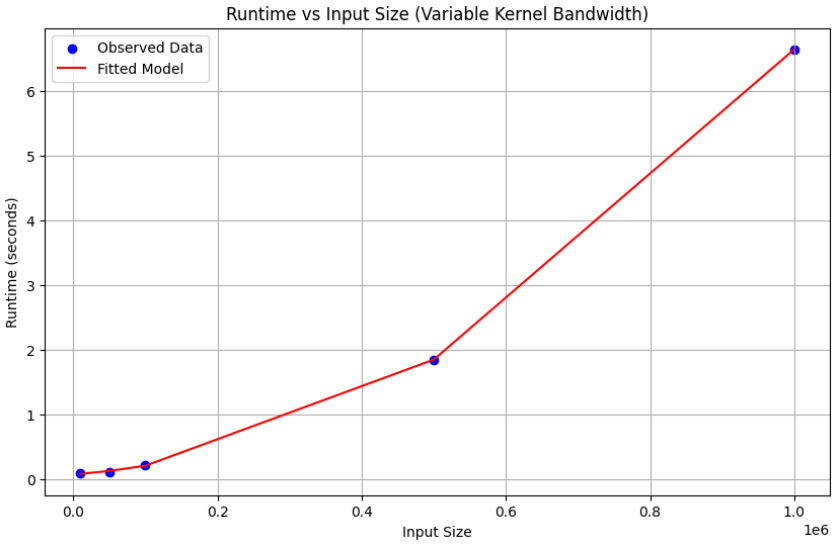
Runtimes for Varying Vector Sizes with a Constant Kernel Bandwidth (0.1, 1):

- Input Size: 10,000 - Runtime: 0.112476 seconds
- Input Size: 50,000 - Runtime: 0.148555 seconds
- Input Size: 100,000 - Runtime: 0.29557 seconds
- Input Size: 500,000 - Runtime: 2.95179 seconds
- Input Size: 1,000,000 - Runtime: 11.0576 seconds
- Input Size: 10,000, Kernel Bandwidth: 1 - Runtime: 0.0908439 seconds
- Input Size: 50,000, Kernel Bandwidth: 1 - Runtime: 0.131137 seconds
- Input Size: 100,000, Kernel Bandwidth: 1 - Runtime: 0.214876 seconds
- Input Size: 500,000, Kernel Bandwidth: 1 - Runtime: 1.83791 seconds
- Input Size: 1,000,000, Kernel Bandwidth: 1 - Runtime: 6.91458 seconds



Runtimes for Varying Vector Sizes and Kernel Bandwidths:

- Input Size: 10,000, Kernel Bandwidth: 10,000 - Runtime: 0.0910181 seconds
- Input Size: 50,000, Kernel Bandwidth: 50,000 - Runtime: 0.11477 seconds
- Input Size: 100,000, Kernel Bandwidth: 100,000 - Runtime: 0.217683 seconds
- Input Size: 500,000, Kernel Bandwidth: 500,000 - Runtime: 1.84758 seconds
- Input Size: 1,000,000, Kernel Bandwidth: 1,000,000 - Runtime: 6.63769 seconds



Observations and Analysis

- There is a consistent increase in runtime with the increase in vector size, regardless of the kernel bandwidth. This indicates a direct relationship between the size of the data and the computational time, typical of data-intensive operations.

- For smaller vector sizes (10,000 and 50,000), the kernel bandwidth (whether 1 or equal to the vector size) does not significantly impact the runtime.
- For larger vector sizes (100,000 to 1,000,000), there is a noticeable difference in runtime based on the kernel bandwidth. Notably, with a vector size of 1,000,000, the runtime is slightly longer when the kernel bandwidth is smaller (1), indicating a more complex relationship between these parameters.
- **Performance on a GPGPU:**
    - The utilization of a GPU appears to offer significant benefits, especially for smaller vector sizes. The ability of GPUs to handle parallel computations efficiently is likely contributing to this performance.
- **Possible Bottlenecks:**
    - For larger vector sizes, the increasing runtime could be indicative of limitations related to GPU memory bandwidth, capacity, or the inherent parallel processing capabilities.
- **Algorithm Complexity:**
    - The increase in runtime with larger vector sizes suggests a complexity greater than O(n^2) but less than O(n^3). The influence of kernel bandwidth on larger datasets hints at additional computational complexity introduced by this parameter.

Conclusion

The algorithm shows efficient handling of smaller to moderately sized datasets on a GPGPU, benefiting from parallel processing capabilities. However, as the data size increases, the runtime increases disproportionately, suggesting a decrease in efficiency, possibly due to limitations in memory bandwidth or the scalability of the algorithm in a parallel computing environment. The kernel bandwidth introduces an additional layer of complexity, particularly noticeable in larger datasets. This analysis provides valuable insights into the performance characteristics of the algorithm, highlighting the impact of both vector size and kernel bandwidth on runtime.

Performance Comparison: GPU vs. CPU

Environment Details:

- System: Dell Inc. PowerEdge R440
- Operating System: Ubuntu 22.04.3 LTS (Jammy Jellyfish)
- GPU: NVIDIA A100
- CPU: AVX512 Supported
- Memory: 16000 MB RAM

Runtimes:

- Input Size 10000, 0.1- Runtime: 1.39847 seconds
- Input Size: 10000, 10000- Runtime: 1.06552 seconds
- Input Size: 10000, 1- Runtime: 1.0698 seconds
- Input Size 50000, 0.1- Runtime: 34.3213 seconds

- Input Size: 50000, 50000- Runtime: 26.8269 seconds
- Input Size: 50000, 1- Runtime: 27.0365 seconds
- Input Size 100000, 0.1- Runtime: 139.798 seconds
- Input Size: 100000, 100000- Runtime: 107.527 seconds
- Input Size: 100000, 1- Runtime: 107.46 seconds
- Input Size 500000, 0.1- Runtime: 3534.57 seconds

Key Observations:

Runtime Efficiency:

- GPU: Shows a consistent increase in runtime with the increase in vector size, but remains significantly faster than the CPU. The efficiency is particularly notable for smaller vector sizes.
- CPU: Demonstrates a much higher runtime compared to GPU, with the increase in runtime being more pronounced as the vector size grows.

On both GPU and CPU, kernel bandwidth has a noticeable impact on runtime, especially for larger vector sizes. However, the GPU handles variations in kernel bandwidth more efficiently.

For vector sizes like 10,000 and 50,000, the GPU shows remarkable performance advantages over the CPU. This is indicative of the GPU's superior parallel processing capabilities for smaller datasets. As vector sizes increase (100,000 to 1,000,000), both GPU and CPU show increased runtimes, but the GPU maintains a relative performance advantage. This suggests that while the GPU is affected by larger data sizes, it still outperforms the CPU.

Potential Bottlenecks:

- GPU: Potential limitations in GPU memory bandwidth, capacity, or parallel processing capabilities become more evident with larger vector sizes.
- CPU: The increase in runtime is more linear, suggesting limitations in handling parallel computations.

Conclusion:

The comparison clearly illustrates the superior performance of the GPGPU over the CPU across various data sizes and kernel bandwidths. The GPU's advantage is most pronounced in smaller to moderately sized datasets, benefiting from its parallel processing capabilities. However, as data size increases, the performance gap narrows, indicating the GPU's limitations in handling extremely large datasets or complex operations. The CPU, while slower, shows a more linear increase in runtime, suggesting its limitations in parallel processing rather than memory or bandwidth constraints.

This analysis underscores the importance of choosing the right computing environment based on the dataset size and complexity of the operations involved. For applications requiring fast processing of smaller to moderate datasets, GPGPUs offer a significant advantage. However, for extremely large datasets or operations with complex parallelism, the limitations of current GPGPU technology become more apparent.