Optimization of Jet Clustering Algorithms for the LHC at CERN

Jonathan Goodrum, Salvatore Rappoccio, Ph.D.
Department of Physics
University at Buffalo, The State University of New York

Introduction

Parallelization is a branch of computing where multiple processes are executed simultaneously. This research focuses on applying parallelization techniques to FastJet, a C++ package used at CERN and other particle accelerators. FastJet provides a wide array of recombination algorithms for reconstructing pp and e+e- collisions. Event reconstruction is computationally intensive, and further advances in the field will ultimately require improved efficiency. Through the use of concurrency, various FastJet algorithms can be run in parallel on multiple processors. By altering FastJet to run on multiple cores, we will decrease event processing time.

What is Jet Clustering?

One of the main focuses in high energy physics is particle collisions

The Large Hadron Collider (LHC) uses software to analyze this data.

Event reconstruction is the process of “rewinding” a collision to better understand the initial reaction. Jet clustering is a step in this process.

Problem Inherent

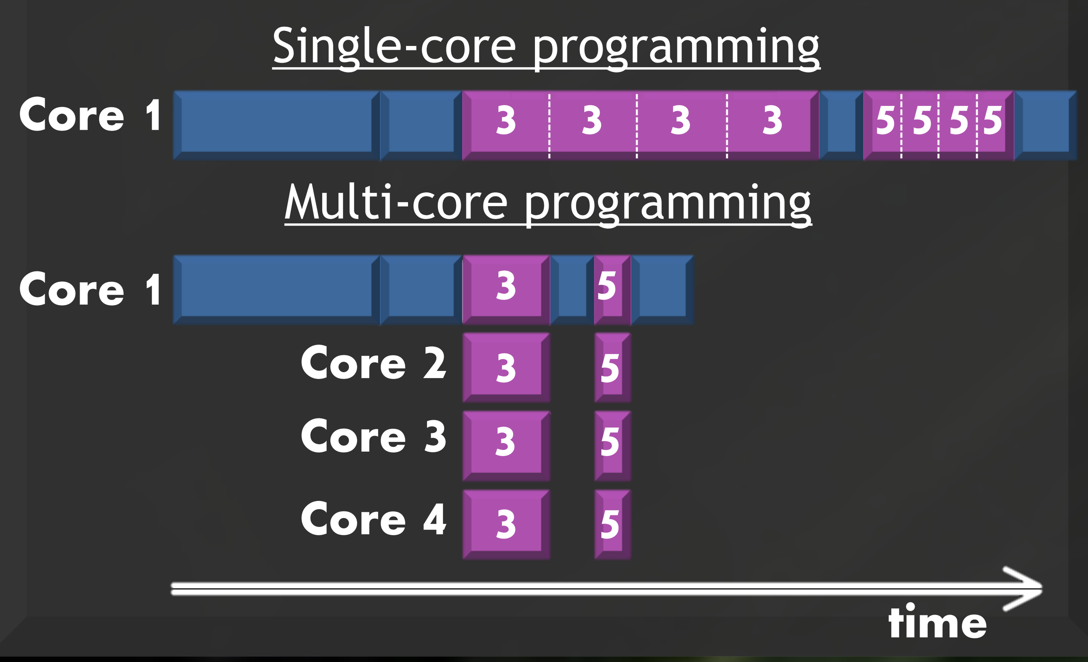
There is a paradigm shift in software engineering, in which concurrency and parallelization play a more fundamental role.

The future of particle physics research, which relies heavily on software and technology, hinges on the development of concurrent programming models.

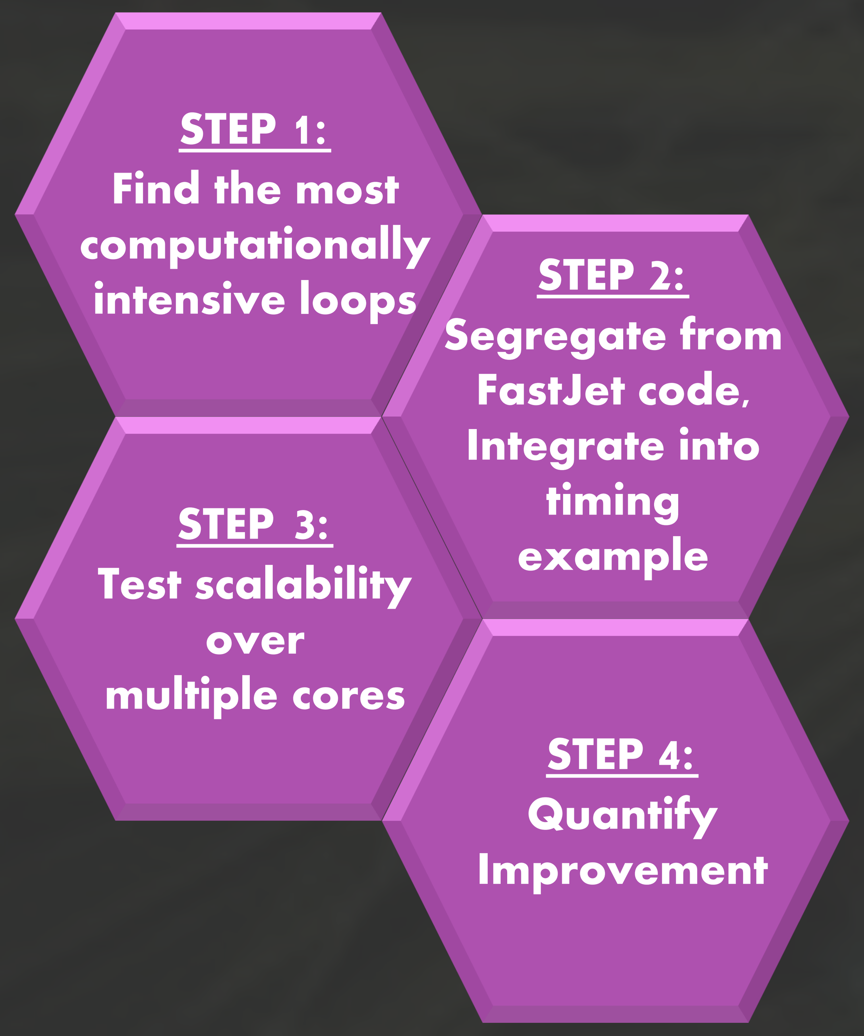
Benefits of Parallelization

In this example, tasks 3 and 5 are order independent, and therefore capable of executing in parallel.

By running these tasks simultaneously on multiple cores, the runtime of this example program is reduced significantly.



Research Methods



- Using concurrent programming, certain operations that are order independent can be threaded to run on several processors.
- To test the scalability of these operations, we would first separate the code from FastJet.
- This section of code is then imported into a timing example. The timing example compares the computation time for the operation on a variable number of cores.
- These steps are then repeated several times for different sections of code within the Tiled N2 jet clustering algorithm.

Results

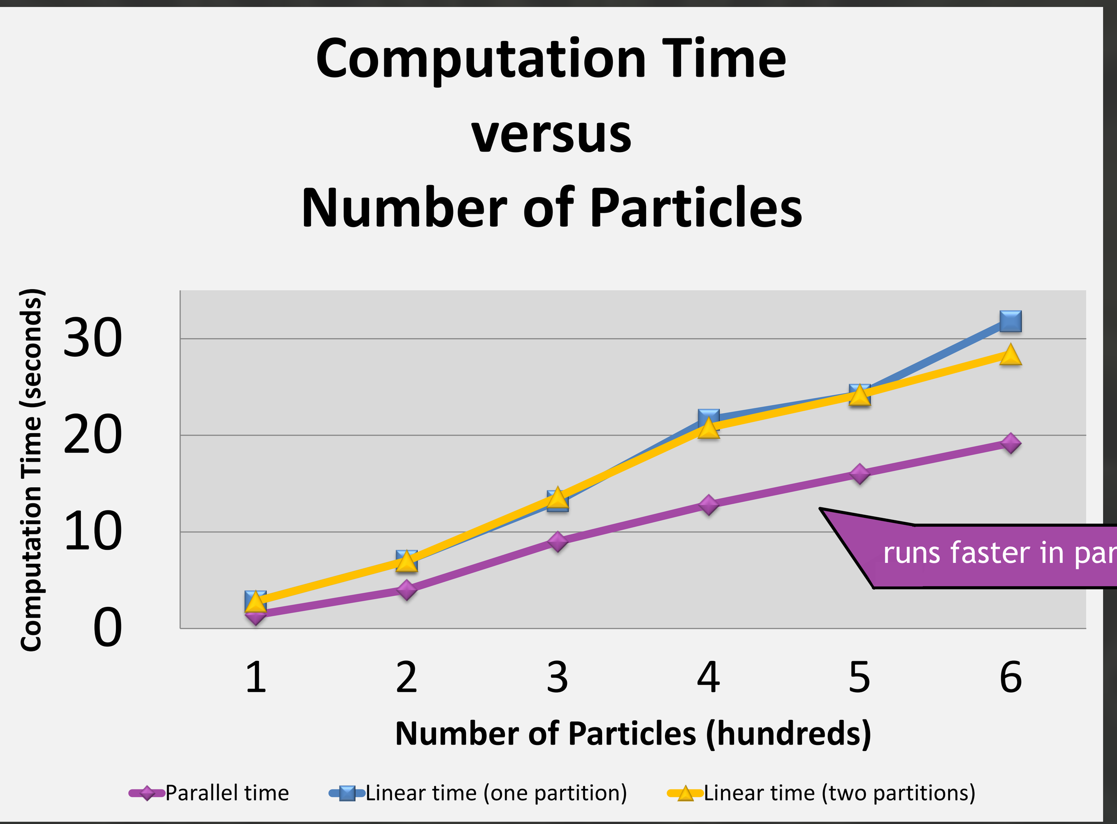


Figure 1: Certain processes in FastJet saw a decrease in runtime when executed in parallel. The increase in speed is relative to the number of particles.

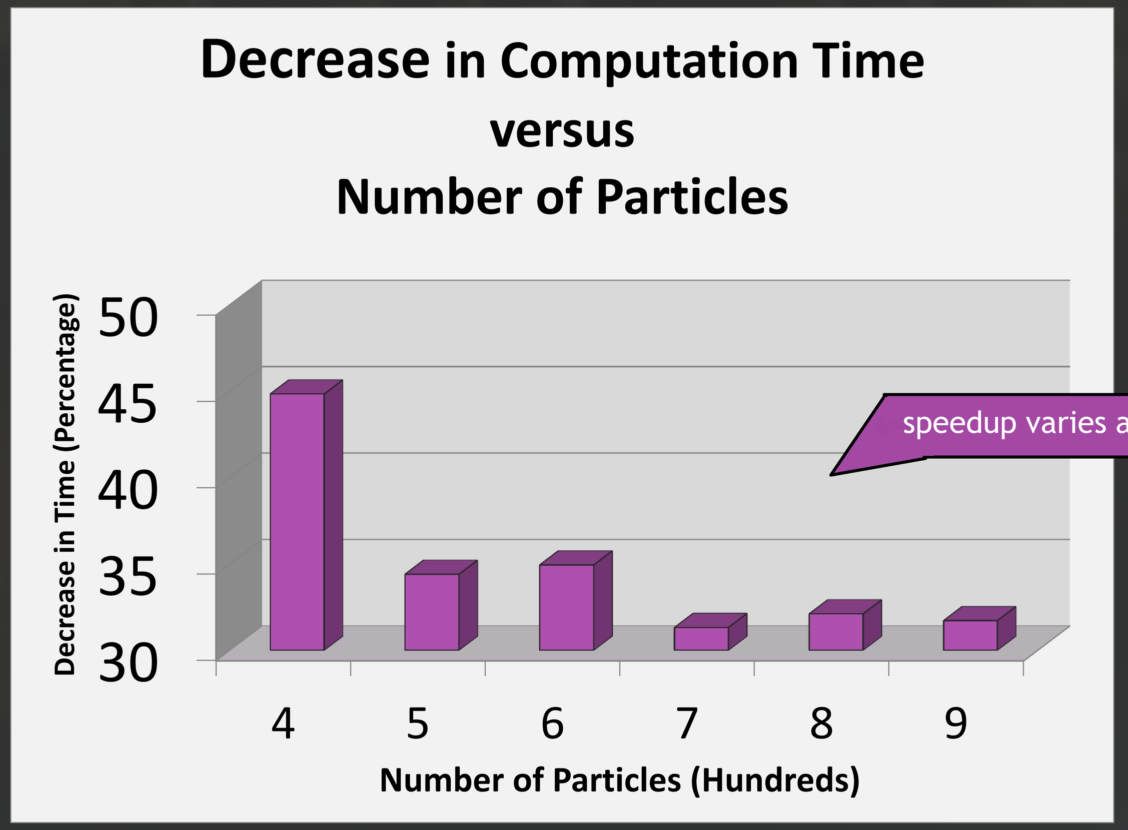


Figure 2: As a percentage of the single-core computation time, the speedup from parallelization varies with the number of particles.

Discussion

- The more intensive a computation, the more it benefits from parallelization

  Certain repetitive processes operate faster on multiple cores, depending on how difficult they are to compute.

  Some processes in FastJet gain a significant speedup when run on multiple cores.

- Less intensive computations actually slow down when run in parallel

  The process of threading a function, delegating its execution to multiple cores, and coordinating the information between these cores is very time consuming.

  Certain processes in FastJet are not intensive enough to merit parallelization, due to the time it takes to thread.

Future Direction

- Focus on memory, not just speed

  The performance of a processor can be defined in terms of runtime and memory usage. Generally speaking, a tradeoff exists when one quality is preferred over the other. Future research should focus on maximizing the benefits of both.

- Optimize other jet clustering algorithms

  Our research used the Tiled N^2 algorithm because it is optimized for a small number of particles (less than 20,000). Future research can focus on other algorithms useful for heavy-ion collisions, such as the N log N clustering sequence.

Acknowledgements

Special thanks to my mentor Dr. Salvatore Rappoccio for introducing me to computing in high energy physics. This work could not be possible without the continued support of Shanna Crump-Owens and the CSTEP program.

References

FastJet 3.0.3 User Manual:

M. Cacciari, G.P. Salam and G. Soyez, Eur.Phys.J. C72 (2012) 1896 [arXiv:1111.6097]