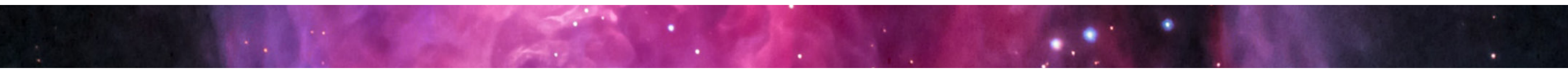
Matching the performance of parallel HACCmk with Parallelware Analyzer

July 8th, 2021



Executive Summary

Scientific simulations on high performance computing systems represent the type of problems where speed is of utmost importance. Even though those systems are fast, large simulations can take days or even weeks to execute. **A way to make simulations faster is parallelization** - distributing the task at hand to several CPU cores.

Our **Parallelware Analyzer helps developers reach the peak performance of their software** through vectorization, parallelization and offloading to accelerators such as GPUs. We wanted to test the impact of Parallelware Analyzer on software performance, comparing the speed-up of the parallel code generated by the tool with that of code parallelized by performance developers on a real world code base.

To measure this, we used the [HACC micro kernel (HACCmk)] which simulates the evolution of the Universe from its early times until present. The parallel version generated by Parallelware Analyzer matched the parallelization performance done by the HACCmk developers.

About HACC

The **Hardware Accelerated Cosmology Code (HACC)** framework uses N-body techniques to simulate the formation of structure in collisionless fluids under the influence of gravity in an expanding universe. The main scientific goal is to **simulate the evolution of the Universe** from its early times to today and to advance our understanding of dark energy and dark matter, the two components that make up 95% of our Universe. HACCmk is the microkernel routine of the code which calculates the short force evaluation.

How does it work?

The short force evaluation kernel is looping over a predefined list of particles and their neighbors and calculates the force values for each particle. The force evaluation for each particle can be independently evaluated, the [current implementation of HACCmk] is **using the OpenMP "parallel for" directive to parallelize over the loop of particles using CPU multi-threading.** The calculation of the force for each particle is implemented as a separate function with particle coordinates, their mass, the list of coordinates of neighbor particles, and their masses, as the input arguments. The force is approximated with a power function over the distance between particles and a 5-th order polynomial [in the reminder.]

The challenge

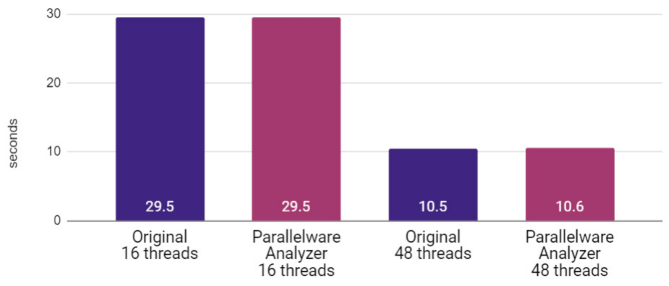
One of the ways to measure the quality of the performance boost achievable using Parallelware Analyzer is to compare it with the code written by expert developers in performance optimization. We took the HACCmk which is already parallelized using OpenMP directives by its maintainers. First, **we established the baseline for comparison by measuring the runtime of the code running sequentially and in parallel using multiple threads.**

In the next step, **we removed existing OpenMP pragmas and ran Parallelware Analyzer on modified HACCmk code base.** Parallelware Analyzer detected a [forall] multithreading opportunity in the crucial loop of HACCmk. Additionally, a [scalar reduction] vectorization opportunity was detected inside the force calculation function. **By executing the commands suggested by the tool we automatically parallelized and vectorized the loops**, respectively.

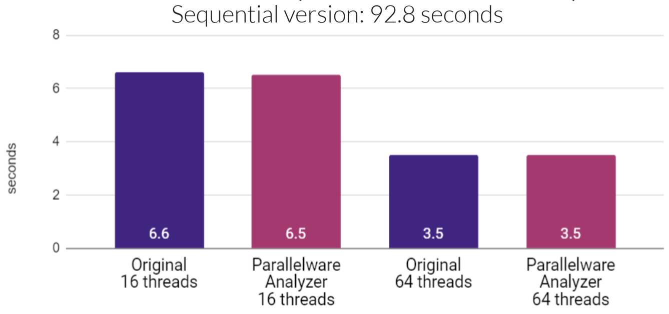
Performance results

We compared the performance of the manually parallelized and automatically parallelized versions of HACCmk and there were no difference in speed. **The version which Parallelware Analyzer parallelized automatically was equally fast as the manually parallelized version.**

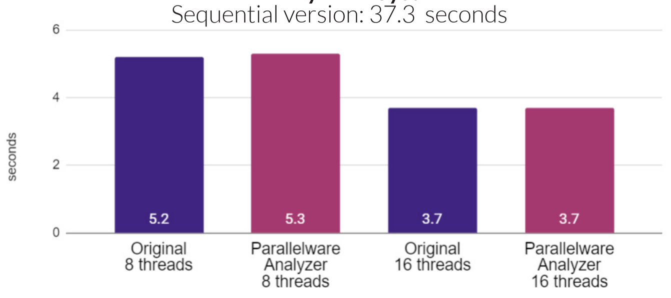
Ookami cluster (ARM-based Fujitsu A6FX node)
Sequential version: 457.2 seconds



EPEEC minicluster (Intel Xeon Gold 5218 node)
Sequential version: 92.8 seconds



AMD Ryzen 7 System
Sequential version: 37.3 seconds



[Parallelware Analyzer] **is the first static code analyzer specializing in performance.** It provides actionable insights through performance optimization reports that help ensure best practices to speedup the code using vectorization, parallelization and offloading to accelerators such as GPUs.

In addition to performance improvements, Parallelware Analyzer **detects defects due to race conditions and data movement issues, verifies the correctness of parallel code and helps enforce best programming practices.** Ultimately, it will help you **speed up the performance of your code while increasing code quality and reducing maintenance costs.**

Thanks to the Stony Brook University and the EPEEC project (Horizon 2020 programme, GA No 801051) for providing access to their systems for benchmarking.

appentra code performance

Appentra is a Deep Tech company that delivers the Parallelware Analyzer software, the first static code analyzer specializing in performance.

www.appentra.com | info@appentra.com | +34 881015556

Unión Europea
Fondo Europeo de Desarrollo Regional
"Una manera de hacer Europa"

FINANCIADA POR ENISA
MINISTERIO DE INDUSTRIA, ENERGIA Y TURISMO