UNIVERSITÉ Grenoble Alpes

Grenoble INP ensimag

Adaptive Parallel Merge Sort in Rust

Quentin Guilloteau

Tuesday 16th April, 2019

Supervisor: Frederic Wagner (LIG)

Context of the Internship

The DATAMOVE team

Areas:

- Distributed Systems
- Parallel Computing
- Scheduling
- HPC

Context of the Internship

The Rust programming language

Rust is:

- Fast ($\simeq$ C)
- Memory safe (Dangling pointers, Buffer overflow ...)
- The future (Slowly becomes a competitor to C++)

The Rayon Library

> *"Data-Parallelism library for Rust that makes it easy to convert sequential computations into parallel."*
>
> — The Rayon Team

Goal of the Internship

Goal

Implement an adaptive parallel merge sort with no added overheads

Utimate Goal

$$\texttt{Rayon} < \texttt{TBB} < \texttt{OpenMP}$$

Goal of the Internship

Goal

Implement an adaptive parallel merge sort with no added overheads

Utimate Goal

$$\texttt{Us} < \texttt{Rayon} < \texttt{TBB} < \texttt{OpenMP}$$

Adaptive Merge Sort

Adaptive Algorithm

An Adaptive algorithm is an algorithm that changes its behaviour at the times it is run, based on information available.

In my case:

- Number of threads available
- Size of the array

Benchmarking - Grid5000

We can actually change some parameters of the algorithm (block size, sorting/fusing policy ...)

$\Longrightarrow$ We needed to benchmark these different versions.

Grid5000

Grid5000 is a large-scale and versatile testbed for experiement-driven research in computer science, with a focus on parallel and distributed computing. ($\simeq$ 12000 cores)

Looking for improvements - Rayon-Logs

Rayon-Logs

Logging extension for Rayon. Allows to visualize the executed tasks of a parallel algorithm (and more !)

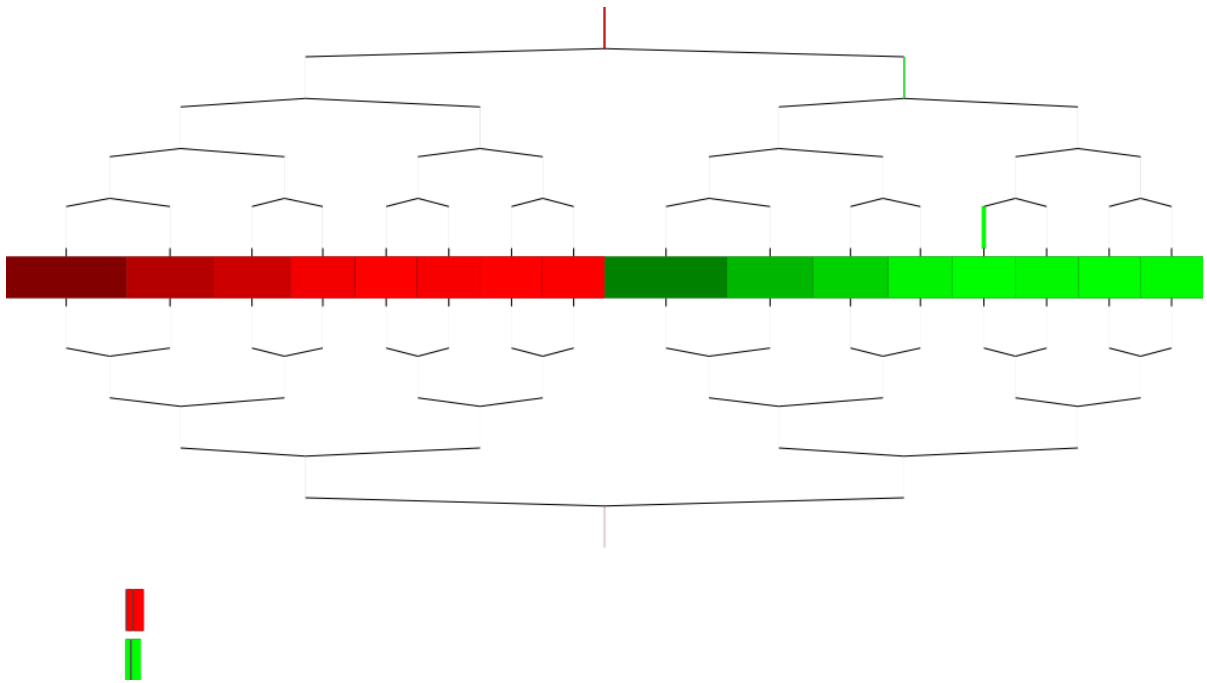

Figure: Example of log for a parallel max

Looking for improvements - Rayon-Logs

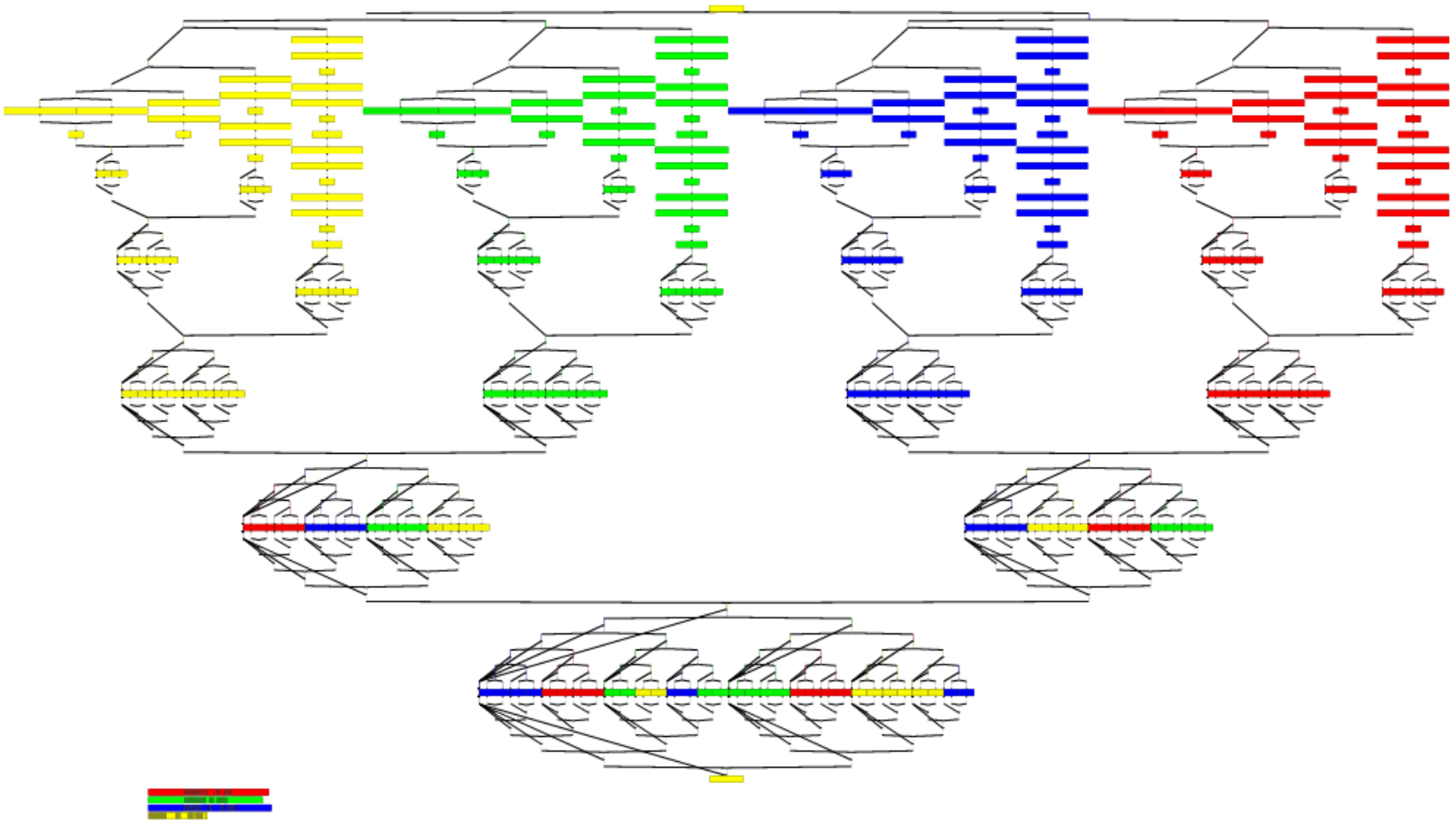

Figure: Example of log for a parallel merge sort

Results that we got so far

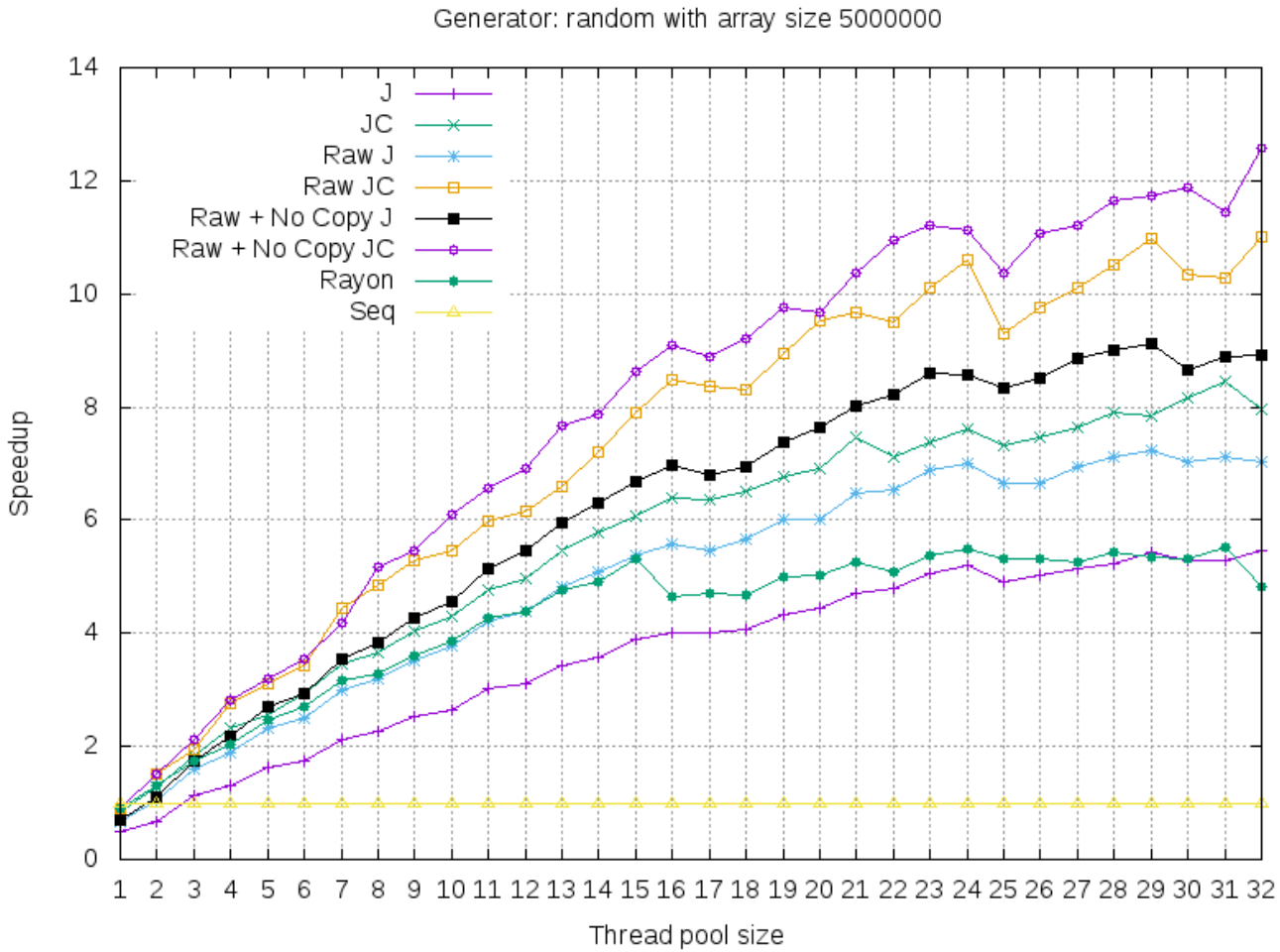

Figure: Speedup on a random vector of size 5 000 000

TODO List

TODO List

- Continue reducing memory usage
- Continue reducing cache misses
- Implement n-ary tasks splitting