MIGRATE: a case study of optimizing of an open source application for estimation of population sizes and gene flow

Jordi Blasco (jordi.blasco@nesi.org.nz)
Sarah Knight (s.knight@auckland.ac.nz)

Outline

Science Background

Estimating migration rates in the budding yeast

There is increasing evidence showing that **microbial populations** are not homogeneous but **structured**.
The processes that drive population structure and connectivity have **implications** for **understanding** the **evolutionary trajectories** of these organisms.

Yeast is of significant commercial importance

Yeast plays a **key role** in the production of **bread**, **wine**, **beer** and other **alcoholic** beverages and is also widely used by the scientific community as a **model research** organism.

Science Background

Regions and analyses of population structure and connectivity

Since microbes are inherently difficult to observe directly, the study relies on **genetic methods to infer their movements**.
Samples were **collected** form vineyards and surrounding native bush in major **Sauvignon Blanc** growing **regions**. The samples were genotyped in **850 individual** isolates at **eight microsatellite loci**[1].

[1] Locus (plural loci) is the specific location of a gene, DNA sequence, or position on a chromosome.

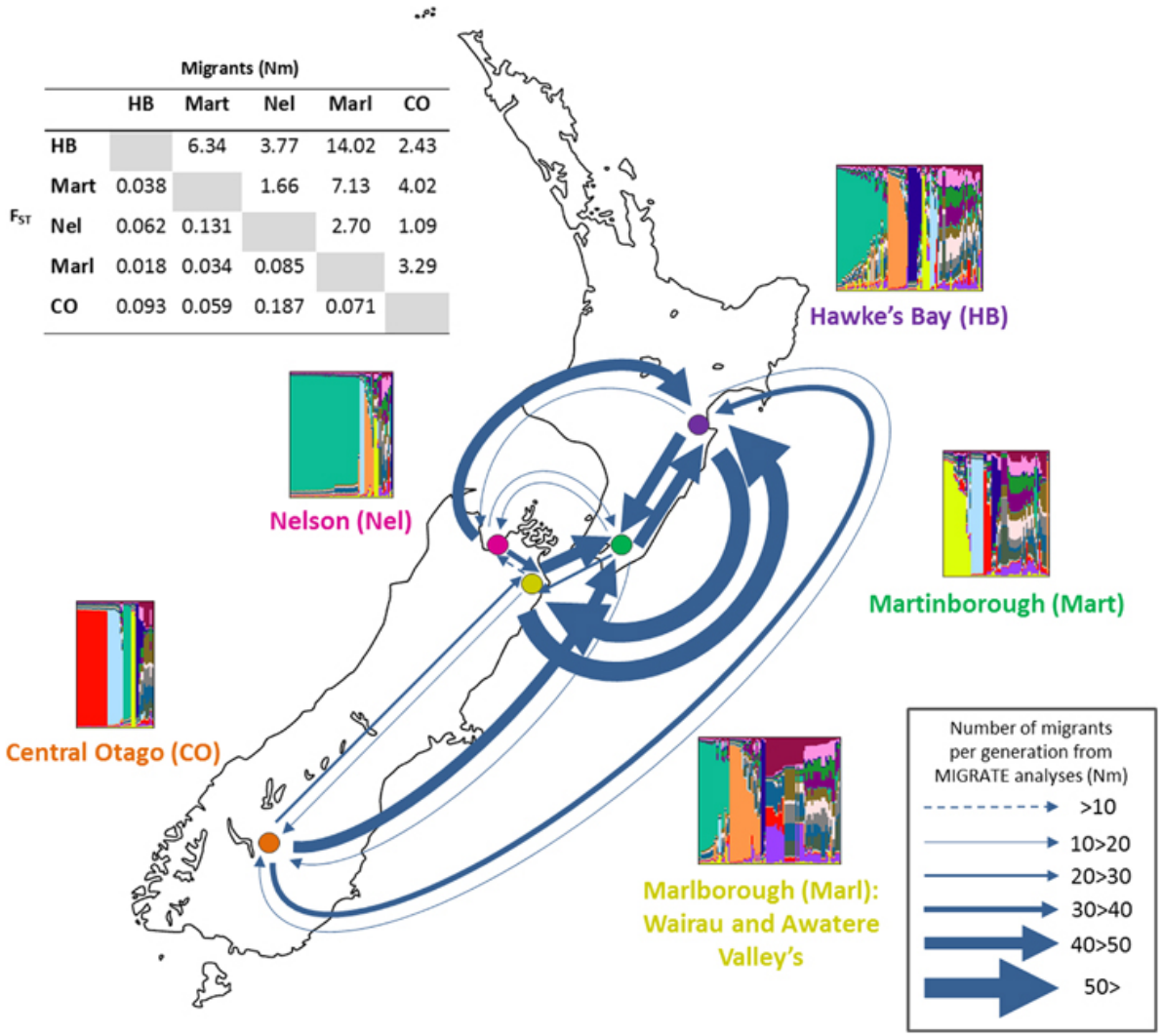


Figure: Source : Knight, S. and Goddard, M. R. (2014). Quantifying separation and similarity in a Saccharomyces cerevisiae metapopulation. ISME J. DOI: 10.1038/ismej.2014.132.

Dimension of the Problem

Evaluating Size Magnitude and Computational Requirements

The dataset was used to infer patterns of population structure and connectivity between these regions and between managed and unmanaged ecosystems within each region.

- **Analyses** of genetic data **to infer migration rates** are typically very **computationally expensive**.
- The final dataset comprised **369 microsatellite profiles**.
- **Bayesian coalescent** approach implemented in **MIGRATE** (Beerli, 2006; Beerli, 2009) had been used.
- **Final run composed of ten replicate MCMC[2] chains of one million steps in length across all eight loci.**

[2] Markov chain Monte Carlo

Domain Decomposition Strategy

Independence of loci

- **Collected loci** are considered **unlinked**, which implies that **each locus** has a **different coalescent history**.
- Each locus can then be run independently from each other.
- Independent runs are combined in a maximum likelihood context by multiplying the likelihood curves.
- The replication option allows to replicate each locus.
- Bayesian inference will run the analysis n times and combine the collected MCMC samples.
- Maximum likelihood (ML) analysis allows combination of independent replicates.

Parallel approach for Domain Decomposition

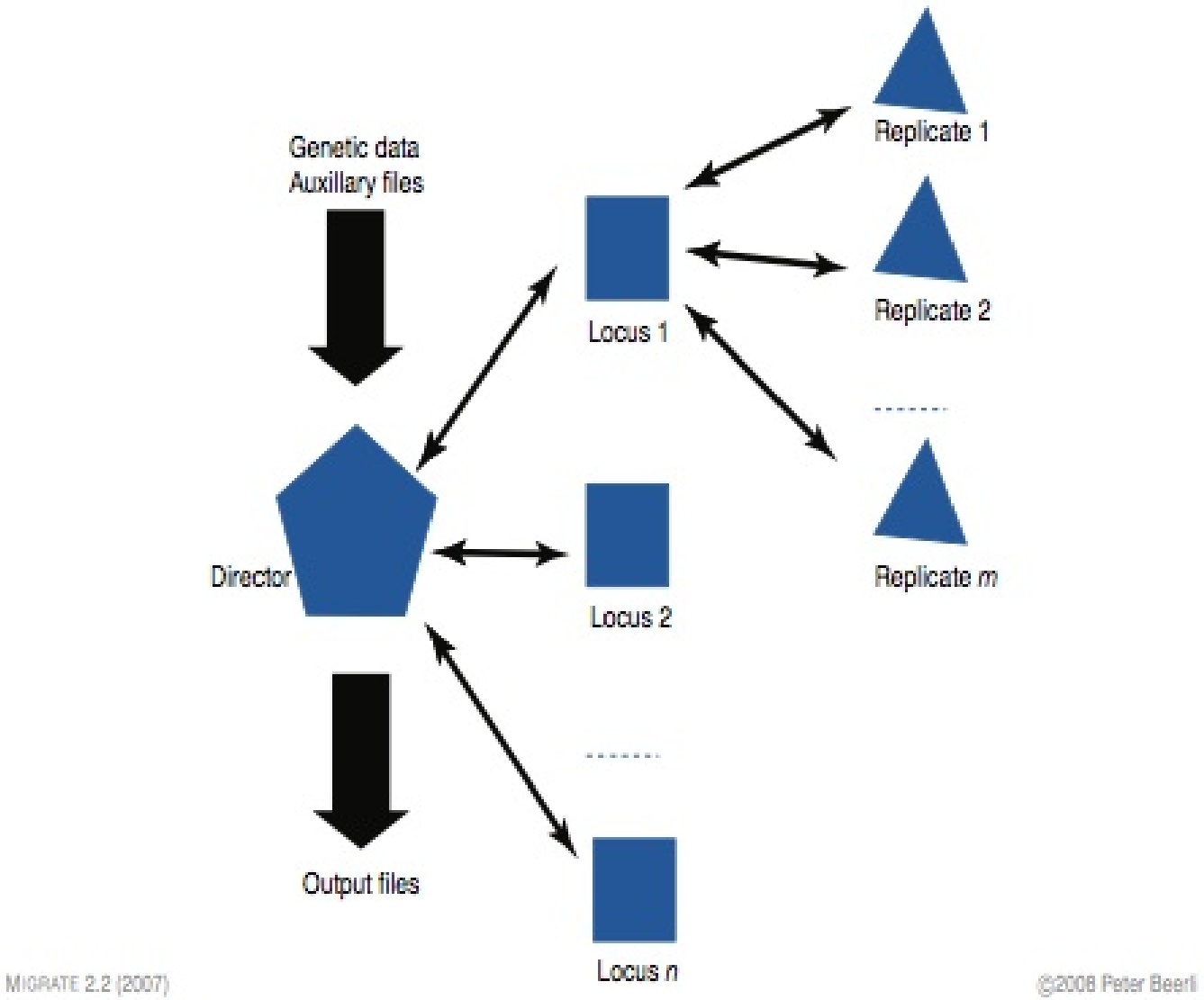


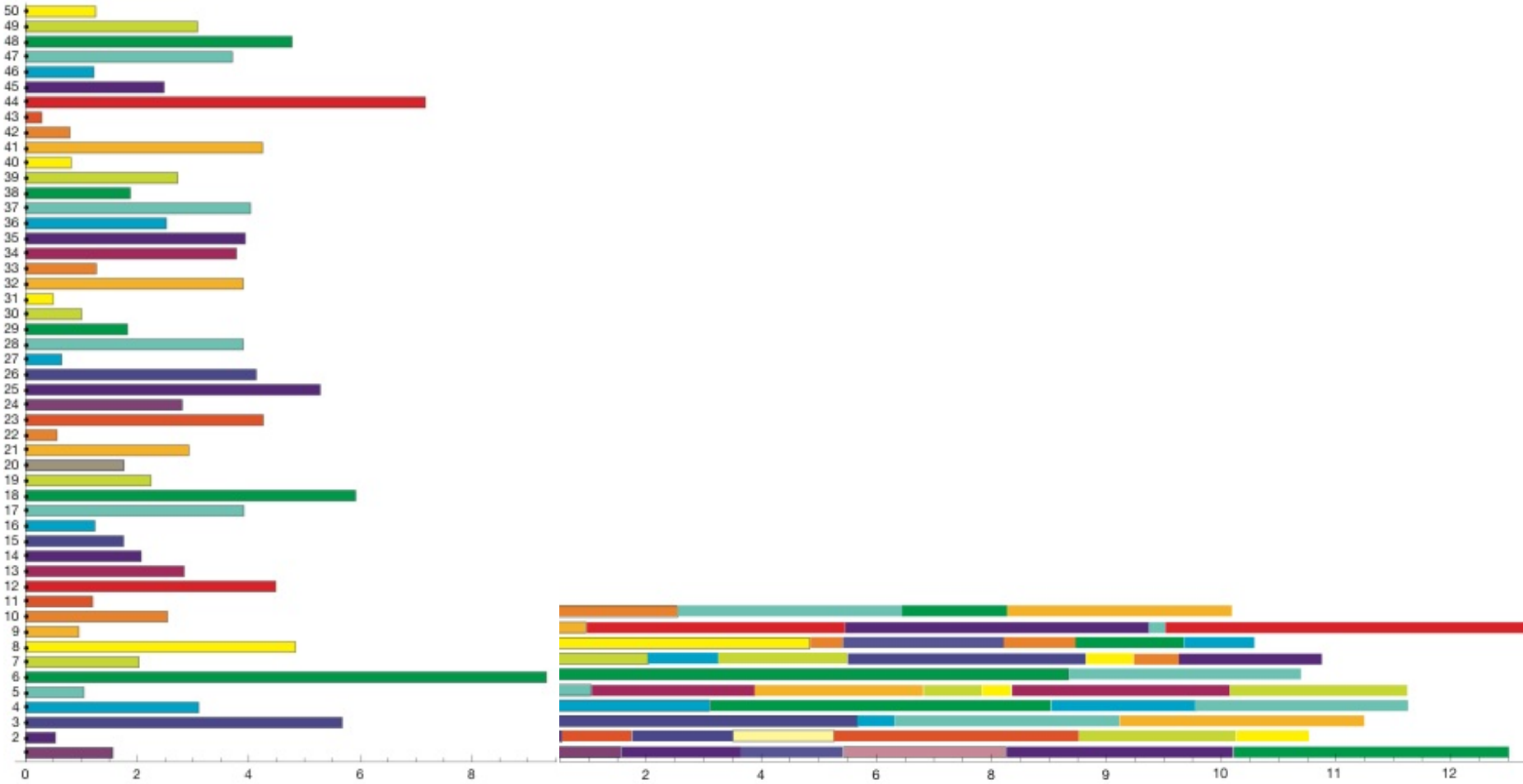
Figure: Source : http://popgen.sc.fsu.edu/Migrate

Domain Decomposition

Improving Parallel Jobs Efficiency

- The runtimes are rather different for each node and the director node will wait for the slowest node.
- In order to improve the usage of HPC resources it is better to allocate fewer nodes than (loci * replicates).
- Running the same problem on less nodes will definitely increase the runtime but also the efficiency.

Parallel approach for Domain Decomposition



(a) 51 cores (b) 11 cores

Figure: Source : http://popgen.sc.fsu.edu/Migrate

Application Performance Analysis

Benchmark details

Sarah Knight provided a representative short test to be used as benchmark, based on real data.

- long-chains=1
- long-inc=1
- long-sample=30000
- burn-in=1000
- auto-tune=YES:0.440000
- replicate=YES:95

Very short walltime (around two minutes using 160 cores).

Application Performance Analysis

Processor Generations

System components used:

Westmere

- Intel X5660@2.8GHz
- 2-socket 6-core
- 2x QPI 6.4 GT/s
- 1333MHz DDR3
- 96GB RAM
- Mellanox Connect-IB QDR
- SSE4 (Streaming SIMD Extensions 4)

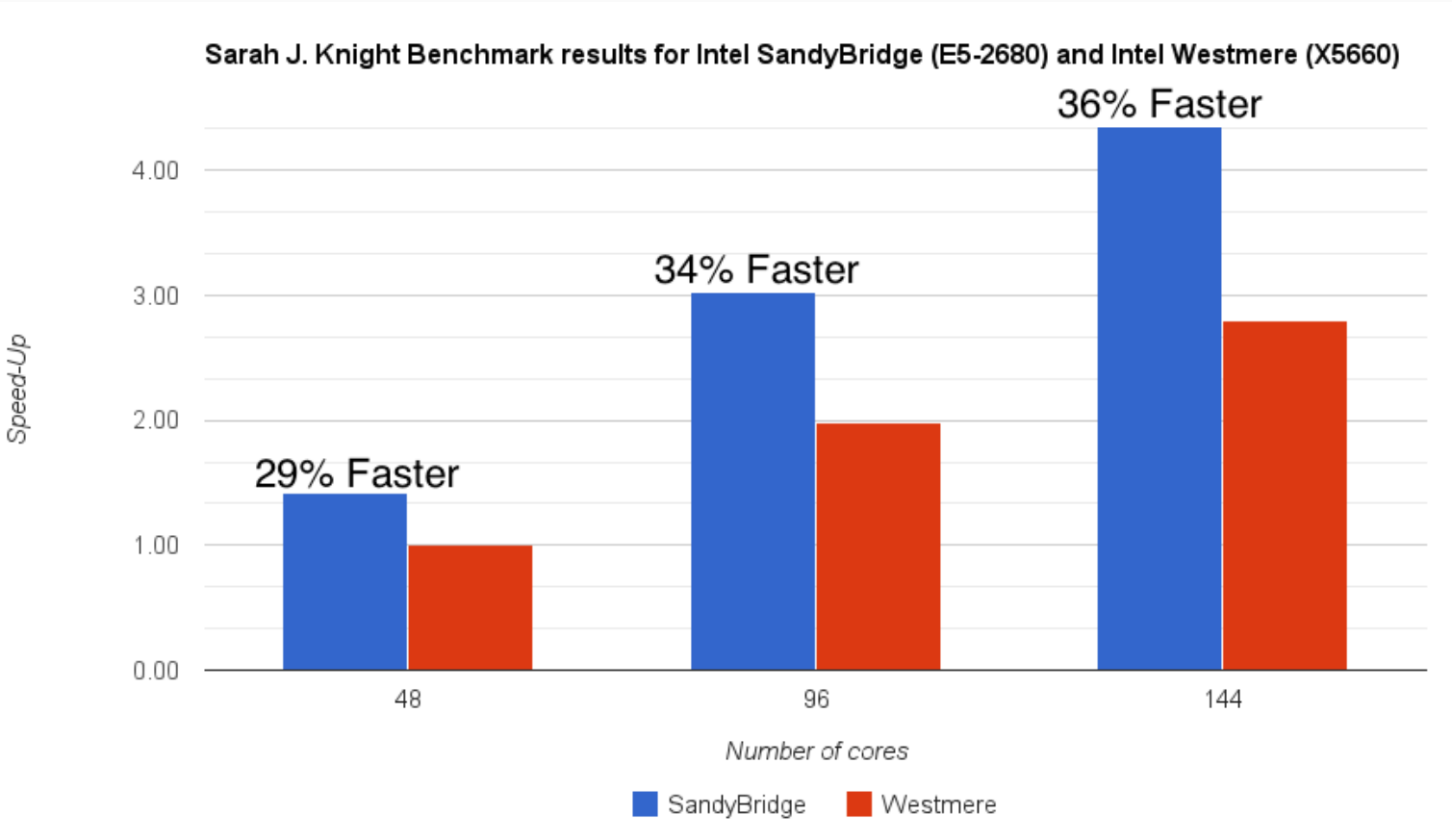
SandyBridge

- Intel E5-2680@2.7GHz
- 2-socket 8-core
- 2x QPI 8 GT/s
- 1600MHz DDR3
- 128GB RAM
- Mellanox ConnectX-3 QDR
- Advanced Vector Extensions (AVX)

Application Performance Analysis

Processor Generations

Intel E5-2680 Series outperforms prior generation. Up to 36% higher performance than Intel X5660 using 144 cores.



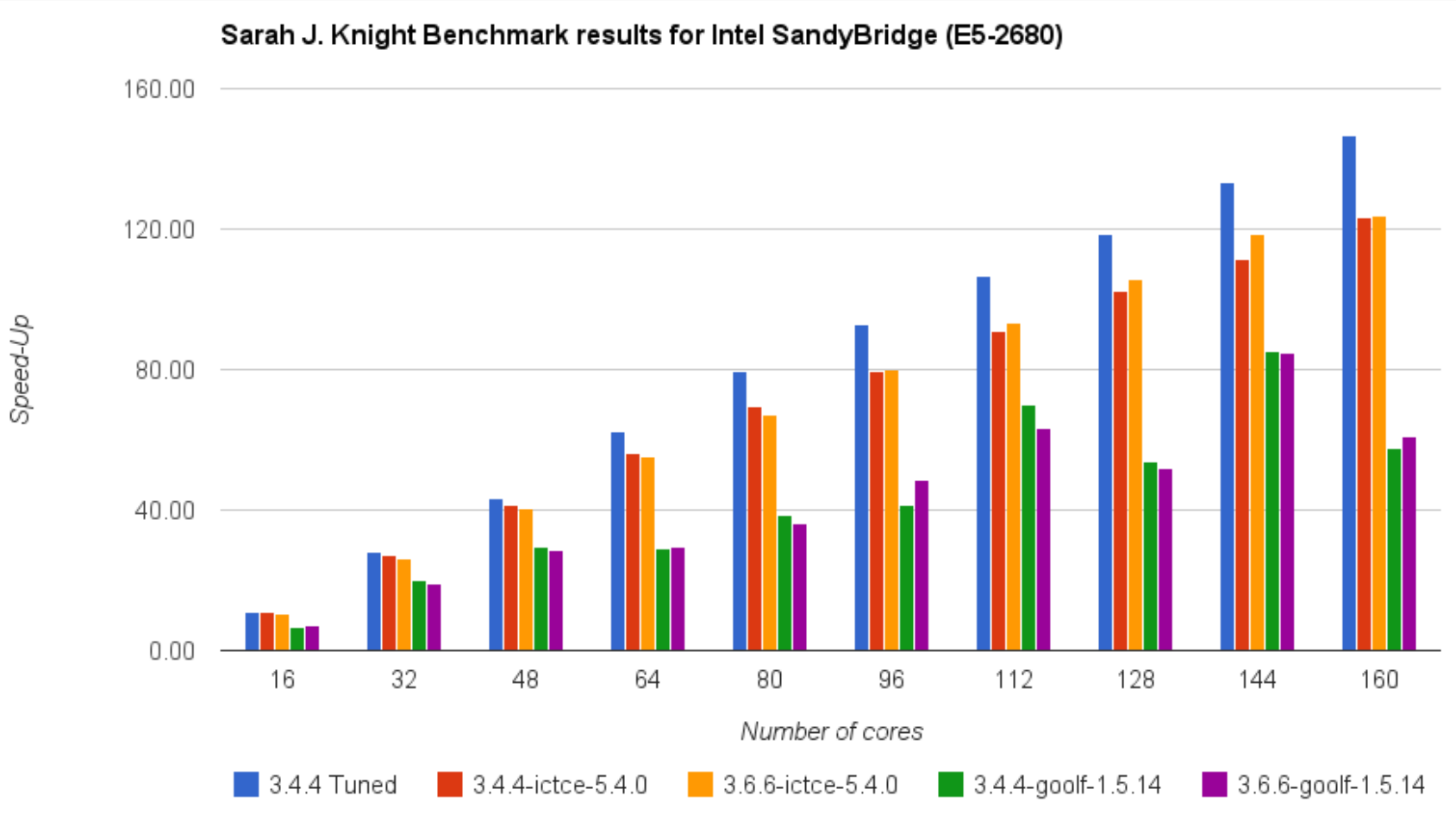
Application Performance Analysis

Compilers

- Using **Intel Composer XE 2013 SP1** compiler shows best performance
- Up to **x2.14 better performance** than compiling MIGRATE using BLACS (0.2.8), FFTW(3.3.4), GCC(4.8.2), OpenBLAS(0.2.8), OpenMPI(1.6.5), ScaLAPACK(2.0.2).
- Instability in the performance observed while running code compiled with GNU toolchain.
- **Tuned version bumps** the **performance** an additional **19%** over best value (default options), x2.54 compared with GNU toolchain version.

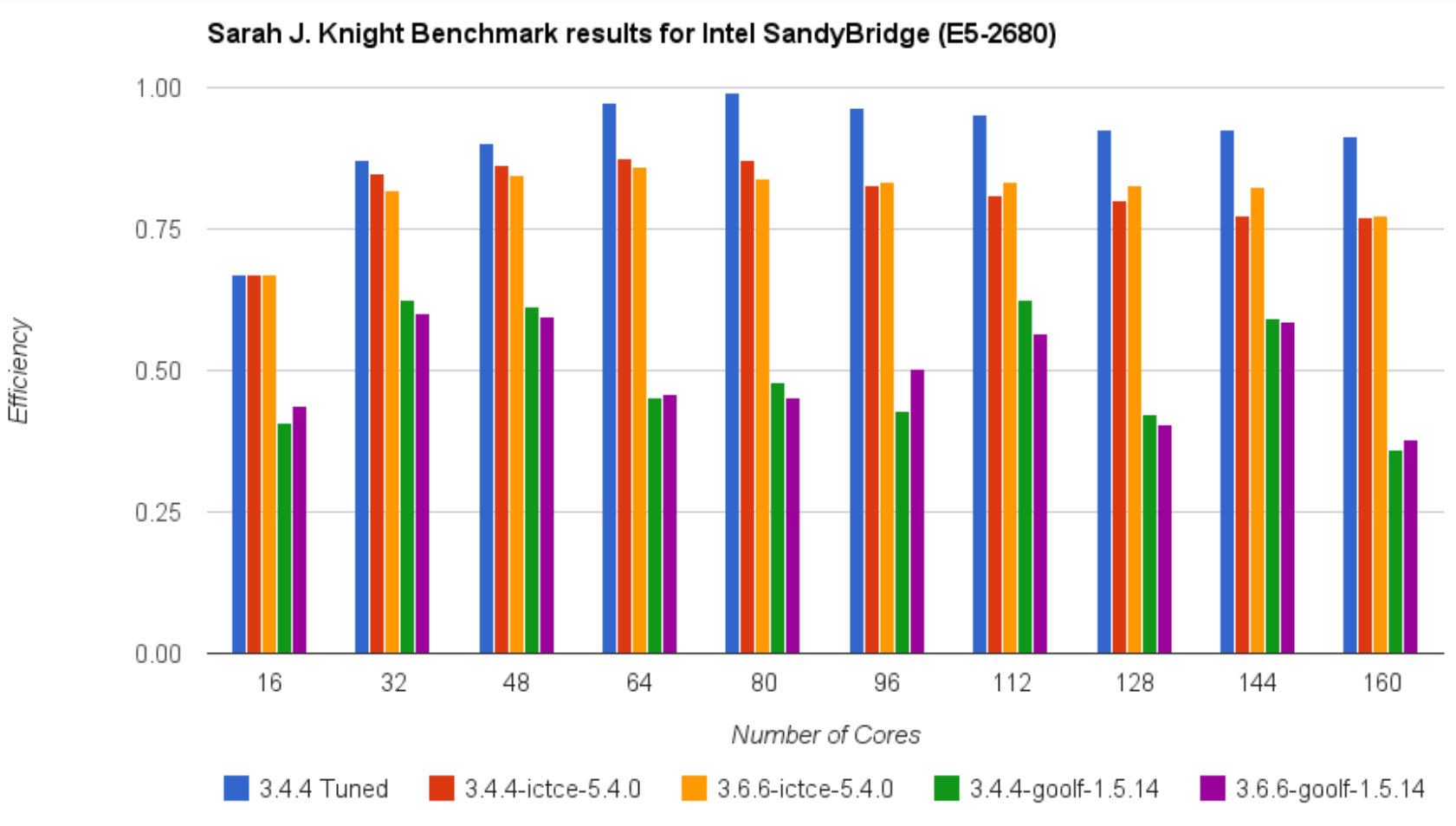
Application Performance Analysis

Toolchains Speed-Up



Application Performance Analysis

Toolchains Efficiency



Application Performance Analysis

MPI Implementations and Options

- Intel MPI performs better than OpenMPI as the job grows in number of cores or the problem scales.
- `I_MPI_FABRICS=shm:dapl` fabrics shows some benefits over MLNX OFED for large number of cores.
- `I_MPI_WAIT_MODE=enable` processes that waits for receiving messages without polling of the fabric can save CPU time.
- `I_MPI_SHM_BYPASS=enable` Turn on/off the intra-node communication mode through network fabric along with shm.
- `I_MPI_SHM_CACHE_BYPASS_THRESHOLDS` Set the message copying algorithm threshold for shm device.
- `I_MPI_INTRANODE_EAGER_THRESHOLD` Change the eager message size threshold for intra-node communication mode.

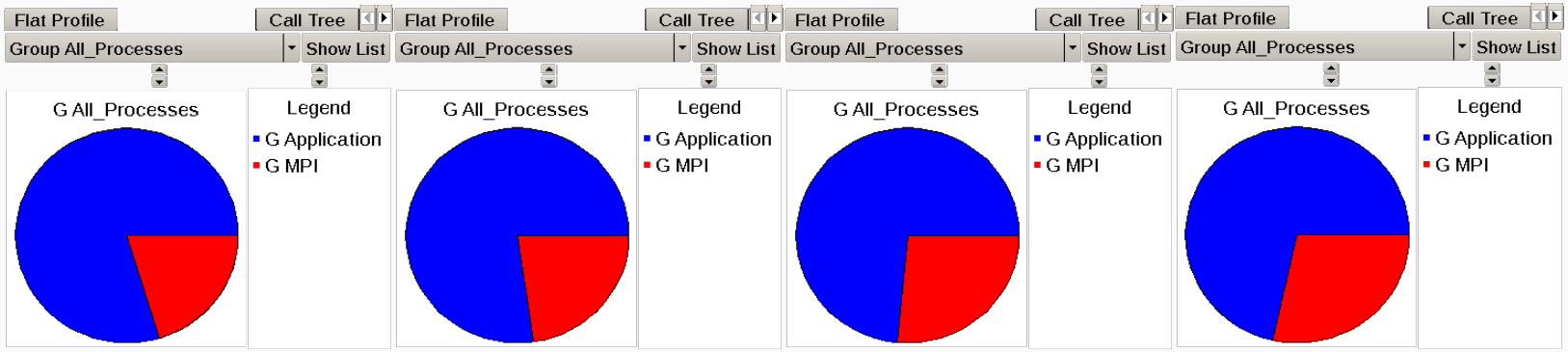
Application Performance Analysis

Communication overhead

$$T = \frac{MessageSize}{BW} + L$$

Time Spent by MPI Calls

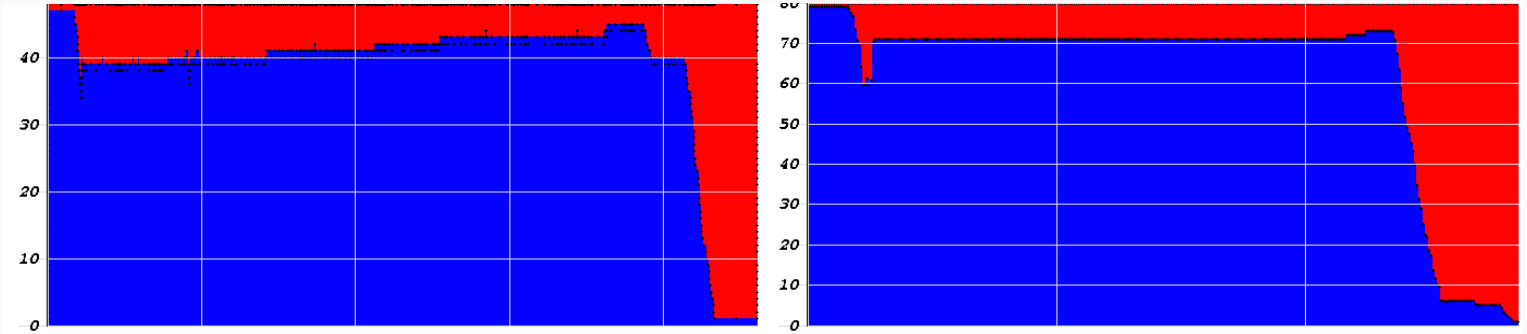
Due to the nature of the code, the major time spent by MPI calls is due to **MPI_Recv**.



(a) 48 cores (b) 80 cores (c) 128 cores (d) 160 cores

Application Performance Analysis

Time Spent by MPI Calls - Quantitative View



(e) 48 cores (f) 80 cores

(g) 128 cores (h) 160 cores

Figure: Number of MPI tasks (y-axis) vs runtime (x-axis). Time spent in MPI (red) grows more than time spent in the application (blue) as the allocation grows.

Application Performance Analysis

Time Spent by MPI Calls - Breakdown View

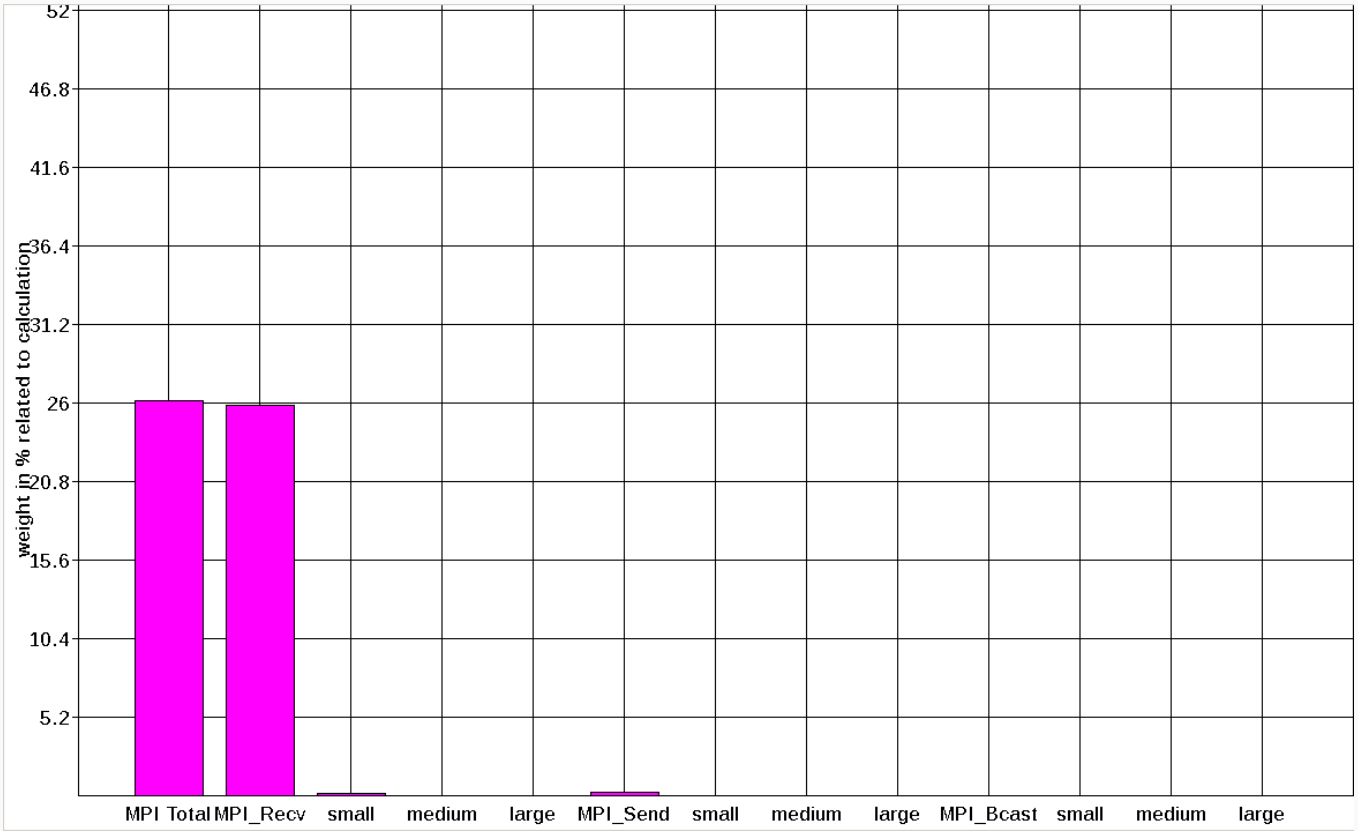


Figure: Time Spent by MPI Calls using 80 cores.**MPI_Recv** is the most expensive operation in this case.

Application Performance Analysis

Number of MPI Calls

- Communication pattern changes as the allocation or the problem scales.
- MIGRATE shows a large number of MPI calls for data communications in the final stage.
- **This number grows dramatically if the number of cores is increased while maintaining the problem size.**

Final Analysis Outcome

Importance of Performance Analysis

- There are theoretical limits but the reality can be really surprising!
- The benchmarks can help to discover the real scalability limits.
- With this information you can get the results even faster and save computational resources for other jobs.
- Intel toolchain definitely improves the performance.
- The code can take advantage of new instruction set available in SandyBridge.
- Intel Trace Analyzer helped to find most suitable values for Intel MPI.

Application Performance Analysis

Major impact in the efficiency and walltime

Thanks to the Application Performance Analysis we have been able to reduce significantly the runtime and improve the efficiency.

Application Performance Analysis

From the original version...

Application Performance Analysis

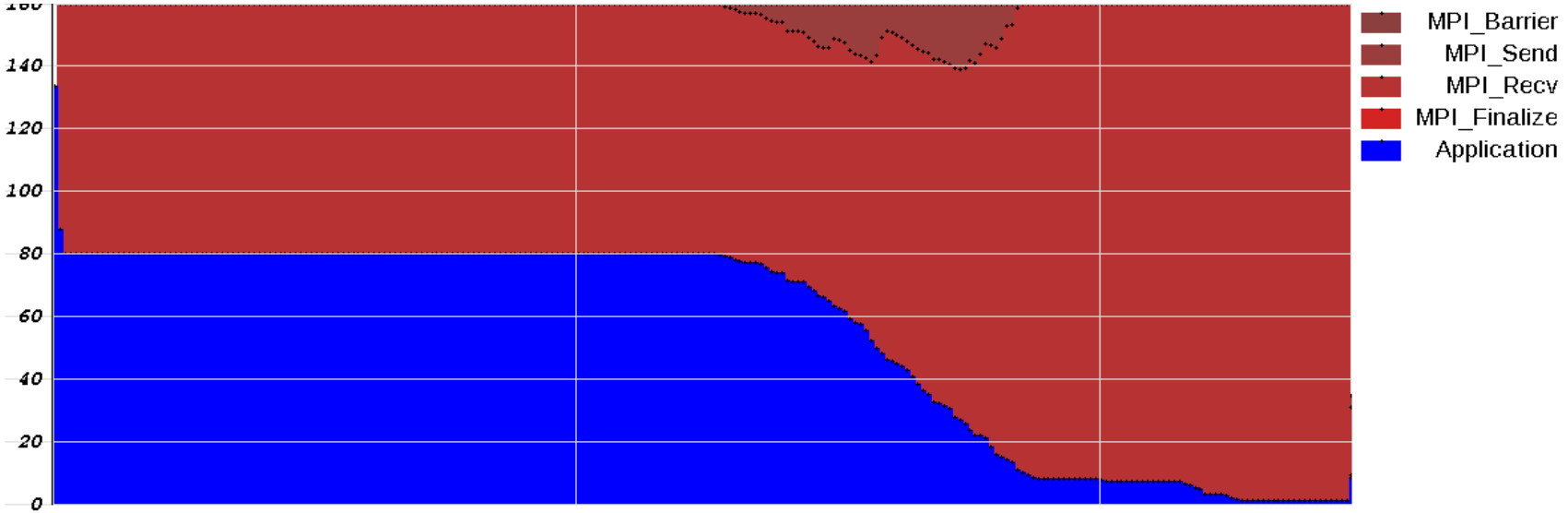


Figure: Time spent by MPI calls (red) using 160 cores versus time spent by application (blue). Efficiency $\approx 38\%$

Application Performance Analysis

... to the production one

Application Performance Analysis

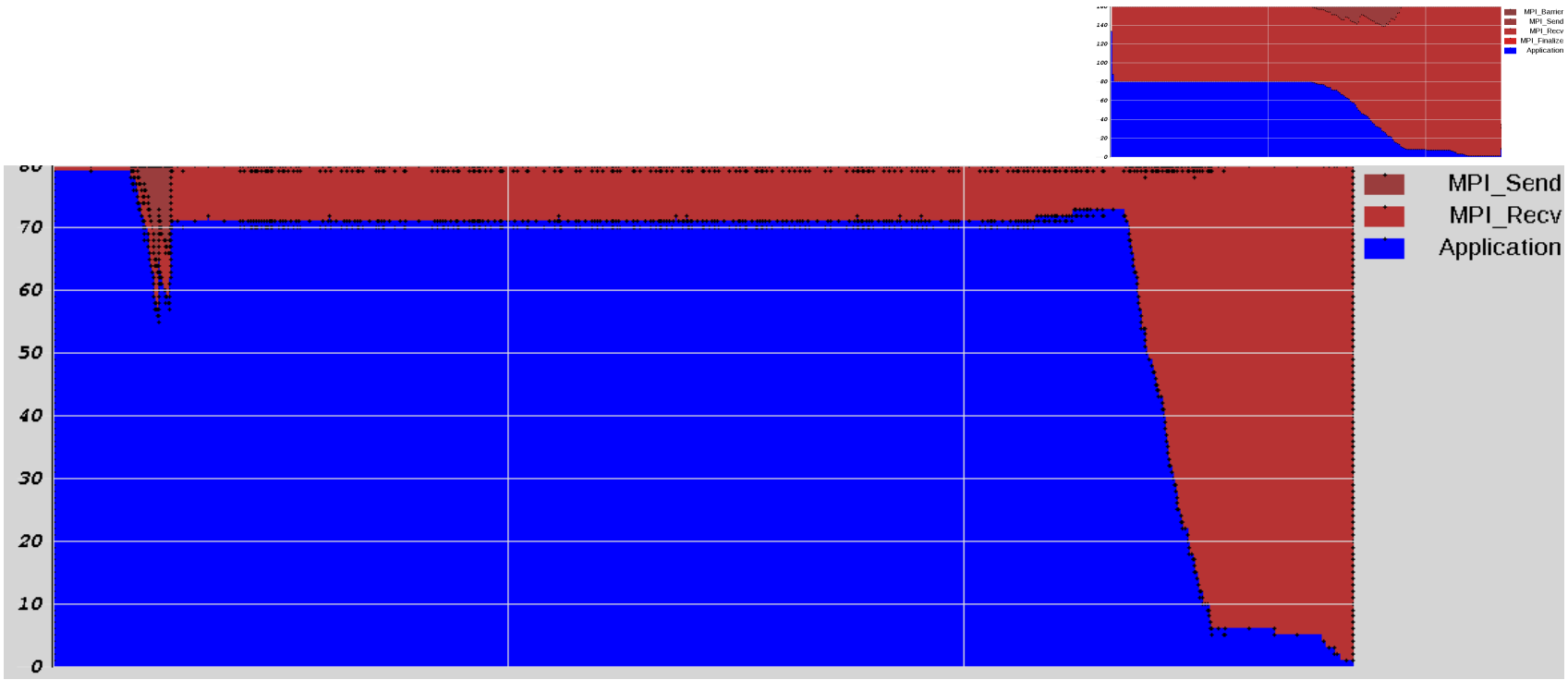


Figure: Time spent by MPI calls (red) using 80 cores versus time spent by application (blue). Efficiency $\approx 77\%$

Final Analysis Outcome

Researcher Feedback

*"**With the help of staff at NeSI**, optimal settings were determined and we managed to run each chain for each locus in parallel, essentially **reducing the time it took to run the analysis by 80 times!** Even then the analysis still took weeks to run and without the resources provided by NeSI this analysis would not have been possible."* – Dr. Sarah Knight (UoA)

The ISME Journal

Multidisciplinary Journal of Microbial Ecology

Knight, S. and Goddard, M. R. (2014). Quantifying separation and similarity in a Saccharomyces cerevisiae metapopulation. ISME J. DOI: 10.1038/ismej.2014.132. `http://www.nature.com/ismej/journal/v9/n2/full/ismej2014132a.html`

Questions & Answers