Genomic Workflow Acceleration on Supercomputers

Kamesh Madduri, Mahmut Kandemir, Paul Medvedev, Padma Raghavan
Computer Science and Engineering, The Pennsylvania State University
sites.psu.edu/XPSGenomics

Project Overview

XPS: FULL: DSD: End-to-end Acceleration of Genomic Workflows on Emerging Heterogeneous Supercomputers, CCF#1439057, Sep 2014–Aug 2017.

The proposed research harnesses parallelism to accelerate the pervasive bioinformatics workflow of detecting **genetic variations**. This workflow determines the genetic variants present in an individual, given DNA sequencing data. The variant detection workflow is an integral part of current genomic data analysis, and several studies have linked genetic variants to diseases. Typical instances of this workflow currently take **several hours to multiple days** to complete with state-of-the-art software, and current algorithms and software are unable to exploit and benefit from even modest levels of hardware parallelism. Most prior approaches to parallelization and performance tuning of genomic data analysis pipelines have targeted computation, I/O, or network data transfer bottlenecks in isolation, and consequently, are limited in the overall performance improvement they can achieve. This project targets **end-to-end acceleration methodologies** and uses **emerging heterogeneous supercomputers** to reduce workflow time-to-completion.

The project focuses on holistic methodologies to accelerate multiple components within the genetic variant detection workflow. It explores lightweight **data reorganizations** at multiple granularities to enhance locality, investigates compute-, communication-, and I/O **task co-tuning**, locality-aware load-balancing, and coordinated resource partitioning to exploit high-performance computing platforms. A key goal of the proposed research is to design **domain-specific optimizations** targeting the massive parallelism and scalability potential of current heterogeneous supercomputers, so that the developed techniques can be easily transferred and applied to dedicated academic cluster and commercial computational environments. Outreach efforts target undergraduate students through recruiting workshops and attract them to interdisciplinary graduate programs. Curriculum development activities emphasize cross-layer parallelism.

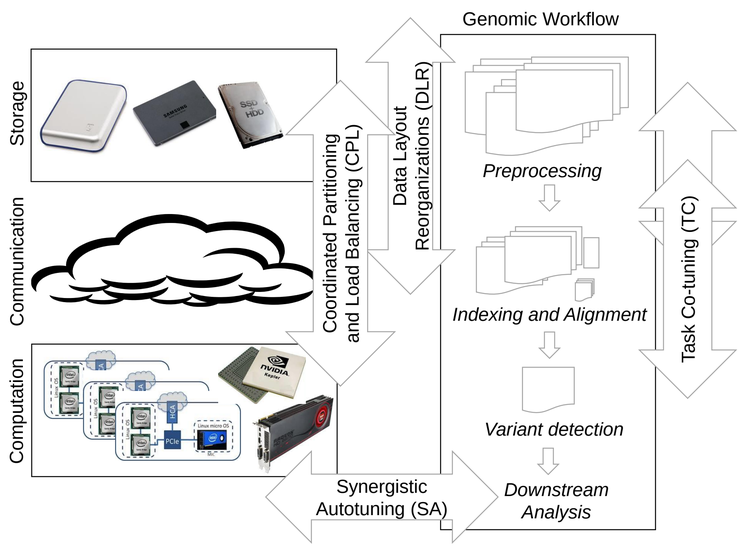

Figure: Research overview: end-to-end pipeline acceleration methodologies, and mapping between the methodologies and the task-specific optimizations.

Task-specific Optimizations	Acceleration Methodologies			
	DLR	TC	CPL	SA
Alignment: Multigrained Parallelization	✓		✓	✓
Alignment: Distributed & Partitioned index	✓		✓	✓
Alignment: Manycore Optimizations			✓	✓
Alignment: SSD-resident index	✓	✓	✓	
I/O-efficient pre-/post-processing	✓	✓	✓	✓
Parallel comparison-based sorting			✓	✓
Parallel variant detection	✓	✓	✓	✓

SPRITE

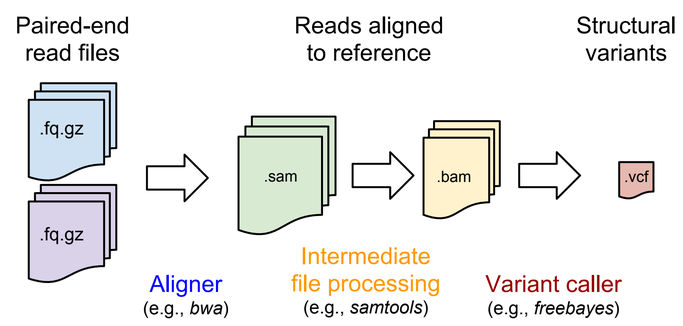

Figure: A simplified view of computational stages in a variant detection pipeline. We indicate popular tools used for each stage and the file I/O formats.

- Single Nucleotide Polymorphisms (SNPs) are the most studied type of structural variation. SNPs are nucleotide differences at a single position.
- SPRITE [1] aims at end-to-end acceleration of the SNP detection workflow, while retaining I/O formats.

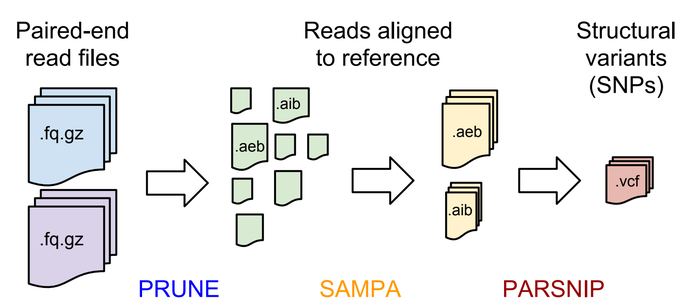

Figure: SPRITE, our HPC pipeline for SNP detection. We design three new parallel tools—PRUNE, SAMPA, and PARSNIP—for various stages of the SNP detection workflow.

- PRUNE is currently based on bwa. For parallelism, the read files are partitioned and the reference sequence replicated on every compute node.
- SAM file creation is optional.
- SAMPA performs a parallel sort of alignment output, prepares data in an intermediate binary format amenable to SNP calling.

PARSNIP

- PARSNIP is a simple counting-based SNP detection tool. It reads SAMPA output to update a nucleotide frequency table $\mathcal{F}$.
- Parallelism through contig partitioning.

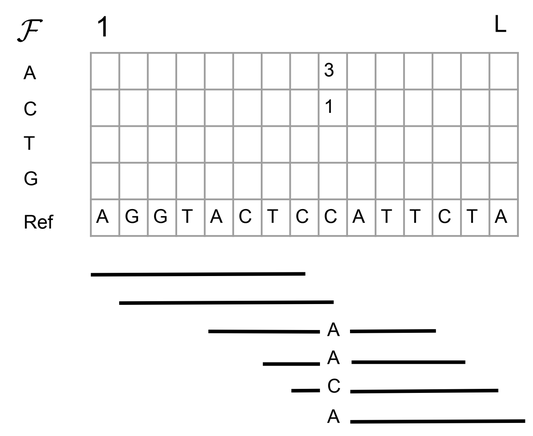

Figure: The organization of the frequency table $\mathcal{F}$ used in PARSNIP. L is the length of the contig being analyzed.

SPRITE and PARSNIP Evaluation

- SPRITE [2] uses MPI+threads programming model.
- We use test data from the SMaSH variant detection benchmarking toolkit.
- Comparisons to a 'reference' pipeline using bwa (v0.7.10-r789), samtools (v1.1), freebayes (v0.9.20).

Performance Results

- SNP calling workflow on SMaSH Venter data set; HuRef SNPs assumed to be the 'ground truth'.
- 16 nodes of NERSC Edison system. Each node has two Intel 12-core Ivy Bridge processors and 64 GB memory. Lustre shared file system with 72 GB/s peak I/O perf.

Pipeline	Ref. Pipeline, 24 cores		SPRITE, 384 cores	
Stage	Tool	Time (min)	Time (min)	Speedup
Alignment	bwa	393	26.36	14.91×
SAM file processing	samtools	401	3.40	117.94×
SNP Calling	freebayes	889	1.55	573.55×
Overall		1683	31.31	53.75×

Table: End-to-end pipeline execution times and speedup. Note that bwa and some phases of samtools are multithreaded.

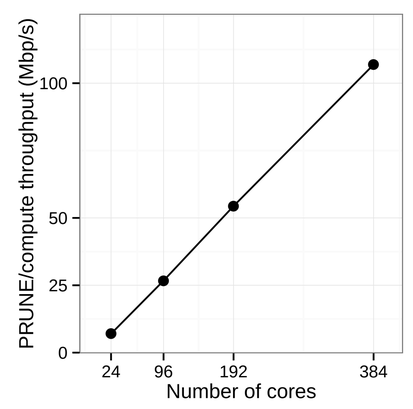

Figure: The compute phases of PRUNE achieve near-linear strong scaling. Inter-process communication is negligible. File I/O accounts for nearly half the overall running time when using 384 cores.

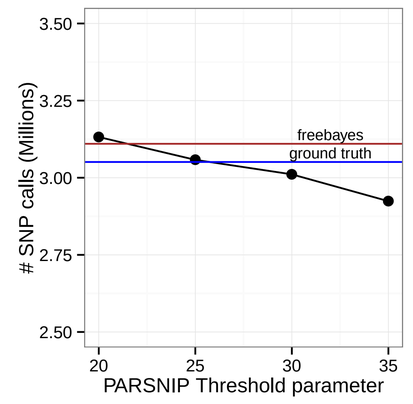

Figure: The total number of SNPs detected by PARSNIP depend on a user-configurable threshold parameter.

Tool	Precision	Recall
PARSNIP	95.1	97.2
freebayes	94.8	97.2
mpileup	98.7	97.0
GATK	99.3	91.7

Table: SNP caller aggregate quality results. PARSNIP results are obtained by setting the threshold parameter to 25.

Future Work

- Tuning SPRITE for alternate hardware configurations.
- PARSNIP GPU and Xeon Phi parallelization.
- Avoiding I/O in intermediate SAMPA step.
- Alternate intermediate and output representations.
- I/O Optimizations in alignment step.
- Alternatives to seed-and-extend alignment.
- Fine-grained index partitioning for alignment.
- Adding probabilistic models to PARSNIP.
- Parallel tools for structural variant detection.
- Lightweight in-memory data layout reorganizations.

References

[1] V. Rengasamy, K. Madduri, Engineering a high-performance SNP detection pipeline, Penn State Computer Science and Engineering Technical Report, April 2015.

[2] SPRITE SNP detection workflow, http://sprite-psu.sf.net, April 2015.

NSF XPS Workshop, Arlington VA, May 31–June 2, 2015