ENGINEERING AT ILLINOIS

CSL: COORDINATED SCIENCE LAB

IGen: The Illinois Genomics Execution Environment

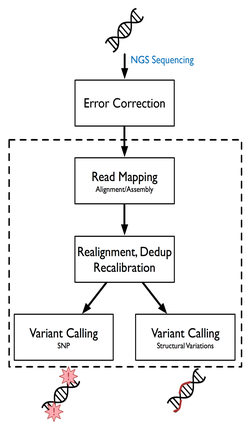
Subho Sankar Banerjee, Ravishankar K. Iyer (Advisor)
University of Illinois at Urbana-Champaign

Goals

- Identify system performance bottlenecks in variant calling workflows running on extreme scale systems
- Identify algorithmic patterns in the variant calling workflow to exploit available parallelism
- Inform the design of future genomics algorithms and their implementations on large computing systems

Application: Variant Calling Workflow

- Identifies and characterizes mutations in NGS data
  - Map NGS data to reference genome
  - Correct for noisy data
  - Differentiate strings in the presence of noise and ploidy
- First phase of the personalized medicine flow
  - Recurrently used
  - Data intensive part of NGS analytics



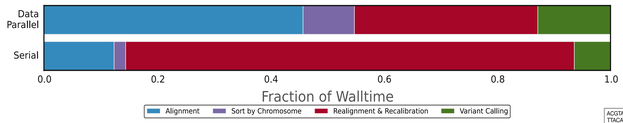
Variant Calling and Genotyping Workflow

- Data filtering process
  - In: 100GB
  - Out: 50 MB
- Best Practices Workflow:
  - BWA for alignment
  - GATK for BQSR and realignment
  - GATK for SNP calling
- Data-parallel distributed computation

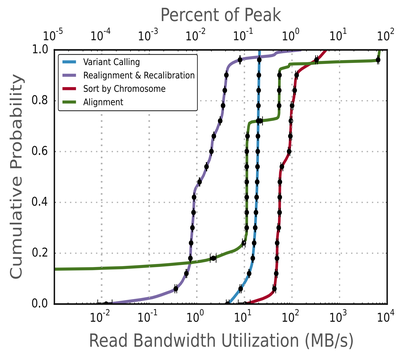
Performance Bottlenecks

Walltime for human genome @ 50x coverage

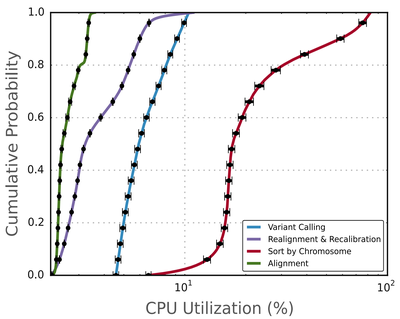
- 43.7 ± 0.01 h on a single node
- 28.3 ± 0.2 h on 22 Blue-Waters nodes



- Very poor resource utilization
- IO dependent performance limitations:
  - File system as distributed memory
  - Htslib: Inefficient for distributed file systems
  - Sorting large alignments on-disk



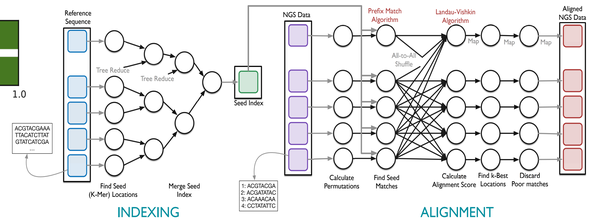
- Compute dependent performance limitations:
  - Serialization across threads in GATK
  - Poor cache locality in Java



Efficient NGS Analytics

Genome Analytics as Data-Flow Graphs

- "Kernels" (Vertices): Data transformations
- "Patterns" (Edges): Data dependencies



Single Ended NGS Read Alignment as a DFG

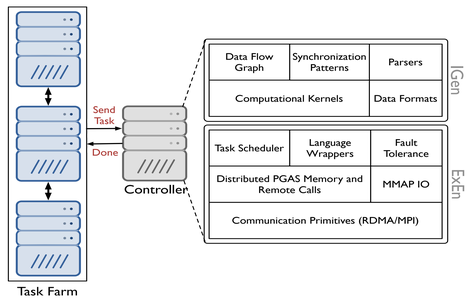
- Implicitly data parallel
- Composable and pluggable model
- Reuse of computational kernels
  - Potential for system level optimizations
  - Potential for accelerators

Workflow stage	Kernels
Error Correction	K-mer computation
Alignment	K-mer computation, prefix tree, Edit-distance computation
Indel Re-Alignment	Edit-distance computation, Table lookup
Re-Calibration	Yates correction, Table lookup
Variant Calling	Entropy, Convolution, Assembly, Edit-distance, Bayesian inference

Common kernels in the Variant Calling Workflow

Distributed DFG executions over heterogeneous clusters

- **Ex**ecution **En**gine: Runtime framework for distributed data-parallel executions of data flow graphs on heterogeneous clusters
- **I**llinois **G**enomics Execution **En**vironment: Accelerated genomic analyses tools for ExEn

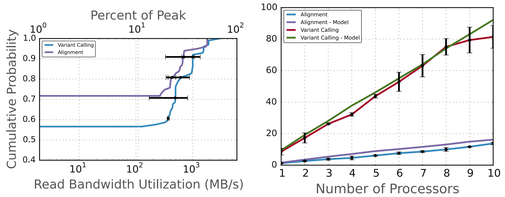


High level view of the ExEn-IGen framework

Performance Enhancements:

- Distributed execution of kernel functions
- RDMA to cut down data-serialization costs
  - RPC: Control Transfer
  - PGAS Memory: Data Transfer
- Efficient data formats
  - Columnar store
  - Use in-memory representation
  - Memory Mapped IO
- High performance kernel functions

Preliminary Results



- Synthetic human chromosome 1 @ 50x
- IGen Aligner (vs. SNAP)
  - Single Node: **1.2x** (35 min to 30 min)
  - Multiple Node (10): **14x**
- IGen Variant Caller (vs. GATK HaplotypeCaller)
  - Single Node: **9x** (36 min to 4.1 min)
  - Multiple Node (10): **81x**

Conclusions

- Measurement driven study of performance bottlenecks in existing NGS analytics tools
- Similar performance pathologies across multiple tools. Scope for system level optimization
- Present a data-flow based abstraction for NGS analytics
- Demonstrate preliminary results of significant performance acceleration
- Simpler to build high performance parts

Ongoing Work

Improved Kernel Scheduling

- Optimal task assignment under constraints of:
  - Affinity
  - Shared resource contention
  - Data Locality

Accelerators

- Explore the use of GPUs for computationally heavy kernels
- Custom hardware accelerators

Deployment Mechanisms

- Containerized deployments using Docker
- Integration with HDFS and Tachyon

Acknowledgements

We would like to thank Zhengping Wang, Varun Bahl, Valerio Formicola, Luidmila Mainzer, Arjun P. Athreya, Zachary Stephens, Zbigniew Kalbarczyk and Victor Jongeneel for their help, support and advice

This research is part of the Blue Waters sustained-petascale computing project, which is supported by the National Science Foundation (awards OCI-0725070 and ACI-1238993) and the state of Illinois. Blue Waters is a joint effort of the University of Illinois at Urbana-Champaign and its National Center for Supercomputing Applications.

This material is based upon work supported in part by the National Science Foundation under Grant No. CNS 13-37732.