IGen: The Illinois Genomics Execution Environment

[Extended Abstract]

Subho S. Banerjee, Ravishankar K. Iyer

Coordinated Science Laboratory
University of Illinois at Urbana Champaign
ssbaner2, rkiyer@illinois.edu

ABSTRACT

There has been a great optimism for the usage of DNA sequence data in clinical practice, notably for diagnostics and developing personalized treatments tailored to an individual's genome. This poster, presents a study of software tools used in identifying and characterizing mutations in a genome. We present IGen, a runtime framework which executes this workflow as a data-flow graph over a partitioned global address space. Preliminary results on the Blue Waters supercomputer show that IGen is able to accelerate single node performance (alignment - 1.2x, variant calling - 9x), as well as distribute the computation across several machines with near-linear scaling. Theoretical models for performance of the entire workflow suggest that IGen will have a 3x improvement in runtime on a single node with near linear scaling across multiple nodes.

Categories and Subject Descriptors

C.2.4 [**Distributed Systems**]: Distributed applications; C.4 [**Performance of Systems**]: Measurement techniques, Performance attributes, Design Studies; J.3 [**Life and Medical Sciences**]: Biology and Genetics

Keywords

Genomics, Variant Calling, Programming Models

1. INTRODUCTION

The prevalence of high-throughput sequencing technologies has led to significant developments in many fields, from plant biology to treatment of human infectious disease, cancer research, and clinical medicine. Variant calling and genotyping is a cornerstone of many genomics pipelines, and is often the most computationally intensive analysis that is performed on the sequence data. The entire workflow can be broken down into the following major components: (1) Error Correction: Correcting sequencing errors in the short DNA fragments called âĂIJreadsâĂİ (2) Alignment: Mapping reads to a reference sequence (3) Realignment & Recalibration: Correcting alignment errors and recalibrating error probabilities based on the alignment; (4) Variant Calling: Identifying and characterizing mutations in the genome by comparing to a population reference sequence. In this work, we explore the performance issues of a popular alignment (BWA) and variant calling tool (GATK HaplotypeCaller), and provide basis for the design of a system which can to perform the same computation much more efficiently.

2. RESULTS AND DISCUSSIONS

To study the performance issues in the variant calling workflow, we collect high-level resource utilization measurements, I/O measurements in the OS, and CPU performance counter events. We perform our experiments on the Blue-Waters supercomputer at the University of Illinois, on synthetically generated[1] human genome data-sets which would be appropriate for clinical use (50x coverage, 0.1-0.4% sequencing error). Our workflow design resembles that of Puckelwartz et. al. [3], in that we exploit data-parallelism in a map-reduce fashion by splitting the input read-sets into multiple parts, computing on the parts separately and combining the results. This allows us to exploit data parallelism available in the problem, but is still incapable to exploiting all the available parallelism. In addition, the single-node performance of many of these tools is quite poor. Our results show that the 99th percentile CPU utilization over the entire workflow does not exceed 10% of the peak and IO bandwidth utilization does not exceed 1% of the peak. The primary cause of this inefficiency is the manner in which these tools perform file IO. All of these tools were written with POSIX complaint file systems in mind, and do not map well on the LUSTRE file system used in Blue-Waters (limitations of common NGS file formats and the utilities used to read and write them). In addition, the internal synchronization mechanism used in some of the tools (especially the GATK based tools) ensure that the multi-threaded execution is serialized.

An algorithmic analysis of the tools being used in the variant calling workflow reveals that almost all the tools in the workflow can be expressed as directed acyclic data-flow graph (DFG) composed of several "computational kernels" as nodes and synchronization patterns as edges. This representation naturally brings out the level of parallelism available in the problem. Though our work is limited to the VC workflow, other work from our research group by Athreya et. al. [1] demonstrated that this hypothesis can be extended to sev-

[1] `http://web.engr.illinois.edu/~zstephe2/read_simulator/`

eral other genomic analyses e.g., multi-sequence alignment, metagenomics, phylogeny etc.

Based on these observations, we propose IGen (The Illinois Genomics Execution Environment), a runtime library which can execute a computational DFG over a cluster of computers. The IGen framework is built on the hypothesis that by optimizing these kernels and the data-flow between them at the system-level, it can be possible to accelerate a large number of computational genomics tools. The framework abstracts the parallel and distributed nature of an application by presenting the programmer with a view of functional execution over a Partitioned Global Address Space (PGAS) over the entire cluster. This significantly simplifies application development because programs can be written for a single machine and run on a large cluster, making it easier for users to build applications for extreme-scale systems. Using this runtime library, we demonstrate the performance improvement of two bio-informatics applications: Single-ended read alignment using the SNAP [4] algorithm and variant calling using the GATK HaplotypeCaller [2] algorithm.

The preliminary results of our experiments show that we are able to achieve a 1.2x speedup (35min to 30 min) for alignment on single node, and a 9x speedup (36 min to 4.1 min) on a human chromosome-1 data set at 50x coverage. And scaling these computations to 10 nodes results in total speedup of 14x for alignment and 81x for variant calling. A straightforward performance model of the kernels used in the workflow and a empirical model of the systems performance based on micro-benchmarks suggests that the entire workflow will see a speedup of 3x on one node and near linear scaling with the number of nodes.

In conclusion, we present a study of the inefficiencies in performance of several bio-informatics tools and design and implement a runtime library which can optimize out several of these inefficiencies at the system level. Thereby showing that it is possible to achieve in some cases a speedup of 81x over the original implementations of these tools.

3. REFERENCES

[1] A. P. Athreya, S. S. Banerjee, C. V. Jongeneel, Z. T. Kalbarczyk, and R. K. Iyer. Decomposing genomics algorithms: Core computations for accelerating genomics analyses. Technical Report UILU-ENG-14-2201, Coordinated Science Laboratory Technical Report, 2014.

[2] A. McKenna, M. Hanna, E. Banks, A. Sivachenko, K. Cibulskis, A. Kernytsky, K. Garimella, D. Altshuler, S. Gabriel, M. Daly, and M. A. DePristo. The genome analysis toolkit: A MapReduce framework for analyzing next-generation DNA sequencing data. *Genome Research*, 20(9):1297–1303, jul 2010.

[3] M. J. Puckelwartz, L. L. Pesce, V. Nelakuditi, L. Dellefave-Castillo, J. R. Golbus, S. M. Day, T. P. Cappola, G. W. Dorn, I. T. Foster, and E. M. McNally. Supercomputing for the parallelization of whole genome analysis. *Bioinformatics*, 30(11):1508–1513, feb 2014.

[4] M. Zaharia, W. J. Bolosky, K. Curtis, A. Fox, D. A. Patterson, S. Shenker, I. Stoica, R. M. Karp, and T. Sittler. Faster and more accurate sequence alignment with SNAP. *CoRR*, abs/1111.5572, 2011.