Experimenting Big Data Applications for Genome Sequence Assembly over NSF-Cloud

Arghya Kusum Das, Praveenkumar Kondikoppa, Seung-Jong Park
School of Electrical Engineering and Computer Science
Center for Computation and Technology
Louisiana State University
Baton Rouge, Louisiana 70803
Email: {adas7, pkondi1, sjpark} @lsu.edu

***Abstract*—In this paper, we discuss about experimenting PGA, a Parallel Giraph-based Genome Assembler that we develop to address challenges involved in large scale genome assembly which recently made its way to the forefront of big data challenges. We juxtapose current state of the art big data analysis software and available supercomputing resources with our assembler that serves as a very good example of both data as well as computation intensive job. Our initial result shows PGA works several magnitudes faster than the contemporary parallel de novo assemblers due to its in-memory graph processing. However, we think its performance is still suboptimal because of hardware resource constraints. We believe that PGA has enough potential to perform 2 to 3 times faster on large scale sequencing data if tested with optimally tuned hardware. Furthermore, the growing size of raw sequencing data imposes a storage and memory challenges in the entire assembly process. This motivates us to experiment our assembler's performance on top of different NSF-Cloud infrastructures like Wisconsin/Cisco or Clemson/Dell clusters. Availability of large storage space per node in TB scale (unlike the Utah/HP clusters with only 120GB Flash) as well as large memory per node make these two clusters a possible solution to the suboptimal performance of our assembler. We discuss the challenges in assembling large human genome on LSU-supercomputing cluster, SuperMikeII to justify our resource requirement and provide a brief intuitive analysis of our resource requirement on these two NSF-Cloud clusters.**

I. Introduction

From a communication-network to a social-network, personalized-medicine to genome-sequencing, large scale graphs are ubiquitous. Efficient processing of these graphs is important for modern data analysis that led the development of several graph processing frameworks in last few years, especially after the introduction of Hadoop, the open source map-reduce framework that became the de-facto standard of distributed computing. On the other hand, the recent advance in massively parallel high throughput dna sequencing instruments, prompted the development of de novo genome assembly. These instruments produce vast amount of high quality short read sequences which generate good quality contigs with high coverage when assembled de novo. Careful construction and efficient analysis of de Bruijn graphs are central to de novo assembly without reference genome. Very few of the existing assemblers can exploit the underlying parallelism of de Bruijn graph analysis. In order to address the challenges in real time large genome assembly, we use Apache Giraph to develop PGA that assembles short reads several magnitudes faster than contemporary assemblers. However, we think PGA's performance is suboptimal and can show huge improvement if tested on hardware tuned for current state of the art big data analytics softwares. The rest of the paper is organized as follows. Section 2 introduces the concept of de novo sequencing and a brief overview of PGA. Section 3 and 4 describes our testing infrastructure and the data size to handle. Section 5 describes current results and shows the challenges in assembling large human genome data (450GB) on one of the LSU supercomputing clusters, SuperMikeII, which justifies our motivation for using NSF-Cloud. Finally, in section 6 we provide an intuitive analysis for our NSF-Cloud requirement.

II. PGA Software Overview

A. De Novo Genome Assembly

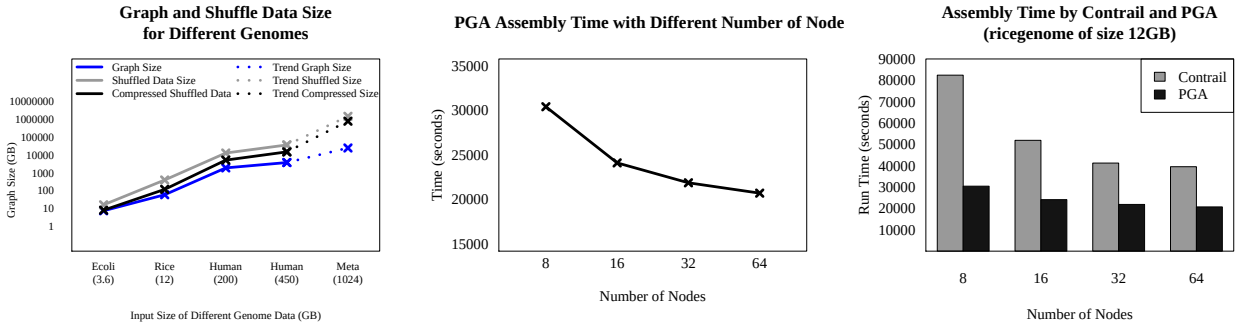
De novo genome assembly refers to construction of an entire genome sequence from a large amount of short read sequences when no reference genome is available. De Bruijn graph construction and removal of read errors (tips and bubble) from this graph is central to de novo sequencing. Finally, resolving repeated regions followed by a scaffolding phase produces long size scaffolds that represents a region in the actual genome.

B. Architecture

We classified de novo sequencing in three different phases. *a*) Building de Bruijn graph *b*) Error removal from de Bruijn graph and *c*) Scaffolding. We store short reads in fastq format in hdfs as input to PGA. In the first phase, we use Hadoop in order to build de Bruijn graph from these short reads. Once the graph is constructed we use Giraph in the subsequent phases to analyze the graph in order to construct appreciably long contigs and scaffolds. We use Giraph for two distinct reasons. *a*) In-memory graph processing performs several magnitudes faster than disk based approach. However, the graph size is limited by the available memory. *b*) Although, the vertex centric, synchronous graph processing model works slower than its asynchronous variation in many cases, it makes the de Bruijn graph analysis extremely easy.

III. Available Testing Infrastructure

We test PGA mainly on SuperMikeII. This LSU supercomputer offers total 382 computing nodes, however, maximum of 128 can be allocated at a time. Each node has two 2.6GHz 8-core Sandy Bridge Xeon 64-bit processors, 32GB RAM and



(a) Size of Data Handled by PGA

(b) Node Scalability

(c) Performance of PGA and Contrail

Fig. 1: PGA Results on SuperMikeII

500GB HDD. However, only 240GB local hard disk space is available as filesystem data storage for Hadoop. The rest is mounted on Lustre. All nodes are connected through 40Gb/s infiniband network with 2:1 blocking ratio.

We also tested some of the large data set (eg. 450GB of human genome) in a 15-node Supermicro cluster with 256GB RAM, 12TB hard disk and 32 cores per node. The high size of RAM, hard disk and higher number of cores per node are exactly in contrast with SuperMikeII. We use it in order to pinpoint the problems and challenges that we observed in SuperMikeII (discussed in section 5b)

IV. Huge Graph and Shuffled Data

Fig. 1a shows the size of intermediate shuffled data and the actual size of the graph generated by the graph building phase of PGA for some of the data sets. The shuffled data clearly indicates very high local disk space requirment, whereas, the actual graph size indicates the huge memory requirement for in-memory graph analysis with Giraph in subsequent phases. It is worthy to mention here that our next plan is to assemble large scale metagenome data. In recent years, high throughput DNA sequencing machines like Illumina genome analyzer frequently produce raw metagenome data that is more than 1TB in size. We show an estimated de Bruijn graph (17TB) and intermidiate shuffled data size (100TB compressed) for 1TB of raw data. This high volume of data and corresponding challenges on SuperMikeII motivates us to explore NSF-Cloud resources in order to find the most optimal one for our purpose.

V. Results and Challenges on SuperMikeII

A. Successful Assemblies

In SuperMikeII, we tested several smaller and medium size genomes including staphylococcus aurious (691MB), Ecoli (3.8GB), human chromosome (10GB), rice (12GB) etc. We also assembled 200GB of human genome data which produces almost 2TB of intermediate de Bruijn graph. We used 128 nodes for this assembly and the entire assembly took almost 10Hrs. In this paper, we selected the rice genome to discuss PGA performance on SuperMikeII. Fig. 1b shows an expected monotonically decreasing time curve with increasing number of nodes which proves the scalability of PGA. Fig. 1c shows a performance comparison between PGA and Contrail, another Hadoop based assembler. The in-memory graph processing with Giraph in PGA clearly outperforms Contrail which uses pure disk-based map-reduce approach to analyze the graph.

B. Challenges on SuperMikeII

As a showcase, we assembled a 450GB human genome data (NCBI website, Accession #SRX016231) with our current infrastructure. Size of the de Bruijn graph for this is 3.8TB. Although, we assembled it in our Supermicro cluster in almost one day, in SuperMikeII we identified two major challenges. First, during build-graph (Hadoop only job), mappers produce huge shuffled data (30TB) which is almost same as aggregate local disk space in 128 SupeMikeII nodes. In a few of the nodes, its size is larger than the available local disk space, resulting in failure of the entire job. Although, compression algorithms like snappy which reduce the size by almost a factor of 3 (as shown in figure 1a) can be extremely useful here, it seems unavoidable while assembling 1TB metagenome data. Second, the build-graph phase produces a huge de Bruijn graph that needs to fit in memory for the subsequent Giraph based graph processing phases.

VI. Possible Solution with NSF-Cloud

A. Wisconsin/Cisco cluster

Each node in this cluster consists of 16 cores, 128GB RAM and 2TB harddisk as mentioned in the website. 450GB human genome produces almost 30TB of shuffled data which means, in terms of storage, we need 15 nodes only. But, the corresponding de Bruijn graph size is almost 4TB that needs to fit in memory. For other OS related work and monitoring, we keep 4 cores and 8GB RAM per core aside which gives 96GB effective memory space per node. So, in order to fit the entire graph in memory, we need almost 43 nodes. Similarly, almost 180 nodes can fit the entire 17TB de Bruijn graph in memory that is estimated for 1TB of metagenome data.

B. Clemson/Dell cluster

This cluster offers 16 cores, 256GB RAM and 4TB hard disk storage per Hadoop-node as mentioned in the website. Keeping 4 cores and 16GB RAM per core aside, we need almost 21 nodes for assembling the human genome and almost 91 nodes for assembling metagenome with PGA.