Amdahl's Balanced Cluster for Data- and Compute-intensive Genome Assembly Application

Arghya Kusum Das[a], Seung-Jong Park[a,*], Jaeki Hong[b], Wooseok Chang[b]

[a]*Louisiana State University, Electrical Engineering and Computer Science, Center for Computation and Technology, Baton Rouge, LA, USA, 70820*
[b]*Samsung Electronics Co. Ltd., 95, Samsung 2-ro, Giheung-gu, Yongin-si, Gyeonggi-do, 446711*

Abstract

The huge growth in sequencing data produced by next generation high throughput DNA sequencing machines forces similar growth in storage and computational resources. The falling sequencing cost is about to turn the computation cost to the dominant one. By applying Amdahl's second law we show that the next generation sequence analytic clusters should be properly balanced in terms of processing speed, I/O (input/output), and memory subsystem to optimize the performance while staying within the budget. We study the applicability of Amdahl's second Law over three different cluster architectures, including a traditional HPC-cluster called SuperMikeII, a regular datacenter called SwatIII, and a novel MicroBrick-based hyperscale system called CeresII, with respect to a genome assembly pipeline. Our study is based on our own benchmark parallel genome assembler (PGA), developed atop current state of the art big data analytic software, Hadoop and Giraph and serves as a real world example of data- as well as compute-intensive application.

Keywords: Amdahl's second law, Balanced cluster architecture, Genome assembly, Big data, High performance computing

1. Introduction

The recent technological advances in the high throughput DNA sequencing machines (e.g., Illumina Genome Analyzer, HiSeq, etc.) revolutionized the genomics study. Per megabase raw sequencing cost is reduced to less than a dollar from more than five thousand dollars fifteen years ago. At this rate, in 2015 the sequencing cost of a whole human genome is reduced to less than five thousand dollar from almost one billion dollar in 2001[1]. Consequently, the growth in the size of the data that is produced by these machines outpaced Moore's Law creating a dire need for a data-intensive scalable solution that can handle this data deluge. At the same time, the hardware cost for storage and processing this big data is increasing linearly that will soon turn the computational cost to the most dominant one [1]. A systematic and general approach is needed for these data-intensive problems both in terms of a computational model as well as cost effective cyber infrastructure.

The best practical approach till date was originally proposed by Jim Gray in 2000 (known as Gray's Laws [2] [3]) where he stated, 1) Bring computations to the data, rather than data to the computations, and 2) The solution is in a scaled out architecture. With the increasing data size, the first law, which basically defines the model of computation utilizing lower network bandwidth, seems to govern the field of genomic analysis for next several years in future. For example, scientists are increasingly using Hadoop and other state of the art big data analytic software which carefully consider data locality. On the contrary, the second law, which defines the underlying distributed cyber infrastructure, although seems to be the most optimal solution apparently, needs to be reexamined. Several studies (e.g., [4], [5], etc.) have been performed recently in this area. However, proposing a high performance, cost effective, and balanced cyber infrastructure still remains a major challenge. We found two major limitation in the previous studies. 1)Most of the studies are limited by the data size that cannot reflect the data explosion of sequencing data. 2)These studies are limited by existing enterprise-standard benchmark workload (e.g., sorting, log processing, social network analysis etc.), thus, cannot reflect the data- and compute-intensive bioinformatics workload.

Motivated by this, we revisited Amdahl's design principals (postulated by Gene Amdahl in late 1960s) over different types of distributed-cyber-infrastructures with respect to a genome assembly pipeline. These design principals are collectively known as Amdahl's second law where he soughted for a balance between the platform's CPU speed, the capacity of main memory, and the I/O bandwidth. These design principals are used as the rules of thumb for designing and evaluating balanced computer systems for decades. In this work, we evaluated three different cluster architectures: 1) a traditional HPC cluster, called SuperMikeII (located at LSU, USA) that offers 382 computing nodes connected with a 40-Gbps InfiniBand, 2) a regular datacenter architecture, called SwatIII (located at Samsung, Korea) that has 128 nodes connected with 10-Gbps Ethernet and 3) a new MicroBrick-based prototype architecture, called CeresII that uses PCIe based communication (also located at Samsung,

*sjpark@cct.lsu.edu
Email addresses: adas7@lsu.edu (Arghya Kusum Das), sjpark@cct.lsu.edu (Seung-Jong Park), jaeki.hong@samsung.com (Jaeki Hong), wooseok_chang@samsung.com (Wooseok Chang)

[1]http://www.genome.gov/sequencingcosts/

Korea). Our evaluation is based upon our own benchmark parallel genome assembler (PGA) which is developed using state of the art big data analytic software, Hadoop and Giraph.

Rest of the paper is organized as follows: Section-2 discusses the back ground and related work of our study. Section-**??** describe the pros and cons of different cluster that we use in our study with respect to Amdahl's second law. Section-**??** describes the PGA software overview. In Section-**??** we present the performance result of our assembler over different clusters and analyze the result with respect to Amdahl's number. Finally, Section**??** concluded our study.

2. Background and Related Work

2.1. Bioinformatics Software on Different Cluster

"Traditional supercomputers focused on performing calculations at blazing speeds have fallen behind when it comes to sifting through huge amount of big data."[2]
Furthermore, downloading the huge genomic data from big genome archival databases (e.g., NCBI[3], EMBL[4], DDBJ[5], etc.) to the HPC clusters for analysis became intolerably time and bandwidth consuming. As a consequence, a rapid migration can be observed in the field of genome research from the existing HPC cluster oriented ecosystem to a cloud based ecosystem which varies significantly in terms of underlying distributed cyber infrastructure. Genome scientists are increasingly using different cloud infrastructures for their data- and compute-intensive applications. Tons of papers (e.g., [6], [7], [8], etc.) has been published in the last few years showing the benefit of cloud infrastructures in the study of genomics. Leading cloud vendors and industrial research labs (e.g., Google, Amazon) started investing a lot in this particular field. For example, Google[6] crowdsourced their software API to store, process and share vast amount of genomics data using Google's cloud infrastructure. AWS (Amazon Web Service)[7] plays a major role in the 1000 Genome project [9], where people can access necessary resources to store petabytes of data and analyze big data genomic pipelines through novel cloud infrastructures.

2.2. Bioinformatics and Big Data Software

Many bioinformatics applications are being written using state of the art big data analytic software. For example, Crossbow [10], harnesses the power of Hadoop [11] to detect single nucleotide polymorphisms (SNPs) in whole-genome sequencing (WGS) data. The BioPig toolkit [12] uses Pig [13] (an analytic software based on hadoop) to support large scale sequence analysis tasks, such as, *k*-mer counting, pathogen detection, etc. ADAM [1] is another high-performance distributed processing pipeline and APIs for the DNA sequencing data based on Apache Spark [14] (an in-memory big data analytic software). The Apache big data analytic software stack has evolved enough in the last few years to address many of the challenges in big data genomic pipeline. However, understanding and designing the hardware environment that these data- and compute-intensive applications require for good performance remain challenging.

2.3. Evaluating Different Cluster Architecture for Big Data Software

From the viewpoint of overall cluster architecture, Michael [5] investigated the performance characteristics of the scaled out and scaled up architecture for interactive queries and found better performance using a scaled out cluster. Their conclusion is allied with Gray's Law [2] as mentioned earlier. On the contrary, Appuswamy [4] reached an entirely different conclusion in their study. They observed a single scaled up server to perform better than a 8-nodes scaled out cluster for eleven different enterprise-level Hadoop workloads including log-processing, sorting, Mahout machine learning, etc.

There are several other studies, which analyze different hardware components for different Hadoop workload. For example, [15], [16], [17], [18], [19] showed the benefit of using SSD (solid state drive) over mechanical HDD (hard disk drive) to accelerate benchmark Hadoop jobs, such as, TeraSort, DFSIO, etc. [20] reached the similar conclusion for Hive and HBase workload also. In particular, all these studies concluded with a storage hierarchy, where the temporary Hadoop data is stored in SSDs and HDDs are used for storing the HDFS data to maintain both performance and cost. This simple conclusion works fairly well for many enterprise level big data workload.

But is it the correct way to measure, especially in the field of genetic research where the input data size is growing at a rate that already outpaced Moore's law with a subsequent increase in temporary data which cannot be accomodated in a single disk? How much benefit a big data genome analytic pipeline can expect from high throughput SSDs in conjunction with the recent processor technologies? In other words, these studies are limited in terms of data size. Consequently, it is not clear, what should be the system designers' considerations to build next generation bioinformatics clusters as they strive for a system balance with respect to processor speed, I/O, and memory bandwidth.

2.4. System Characterization using Amdahl's Second Law

System characterization studies for data centric computing was started with the pioneering work of Jim Gray [2] in 2000. Bell [21] also realized the importance of Amdahl's second law to build balanced system for data-intensive computing. Szalay [22] proposed a balanced system called GrayWulf for data intensive computing that won the storage challenge in SuperComputing 2008. In a later study [23] Szalay proposed another alternative architecture for energy efficient computation using Amdahl's bananced blades. Cohen [24] applied Amdahl's second law to study the interplay between processor architecture and network interconnect in a datacenter. In a recent work,

[2] http://spectrum.ieee.org/tech-talk/computing/hardware/ibm-redesigned-supercomputers-to-solve-big-data-problems
[3] http://www.ncbi.nlm.nih.gov/
[4] http://www.ebi.ac.uk/
[5] http://www.ddbj.nig.ac.jp/
[6] https://cloud.google.com/genomics/
[7] https://aws.amazon.com/1000genomes/

Liang [25] applied Amdahl's second law for performance characterization of two different state of the art big data analytic software (e.g., Hadoop and DataMPI).

3. Motivation: Architectural Imbalance in Different Hardware Components

3.0.1. Network architecture

Traditional HPC clusters uses a hierarchy of switches with a fat tree model of connectivity. This approach typically uses layered architecture. The hosts or servers are connected to the bottom layer, called access layer which is then aggregated to an intermidiate layer of switches, called edge layer or leaf. The edge layer switches are then connected to the top layer switches, called core layer or spine where the actual bandwidth of the network is scaled. To prevent oversubscription, the link speeds got progressively higher from access layer to leaves to spine starting from only few Mbps to several Gbps. This architecture may suffer from bottleneck issues thereby introduces architectural imbalance in the modern datacenter or cloud infrastructures as the bandwidth of the host adapter is increasing in an outstanding pace (an observation by Cohen [24]). Furthermore, to accomodate many servers with fewer switches (thus, reducing the cost of the network) a blocking is used which again limits the performance of big data genomic applications which demand more bandwidth.

As an alternative, the simple Clos based architecture is gaining popularity where all the lower layer switches (i.e., leaf) are connected to all the top layer switches (i.e., spine) using a full mesh topology, thus achieving a non-blocking model using inexpensive devices. The data-intensive applications based of Hadoop or Giraph, which transmits huge amount of data over network in different phases of computation, can use more bandwidth from this all-to-all connection model.

3.0.2. Storage architecture

The performance of any system is traditioanlly constrained by the I/O subsystem. So is the traditional HPC cluster which are normally equipped with only one hard disk drive. Over the span of last 10 years the the hard disk speed is improved by only twice from 7000RPM to 15000RPM. Consequently, the I/O throughput increased from 80MB/s to 150MB/s. At this rate, to read just a 1TB hard disk takes almost two hours. In real world scenario, where a huge amount of data is read/write from/to the local disk the performance may be even worse. So, in many state of the art data center cluster and cloud infrastructure the data is stripped across multiple smaller disks to improve the I/O throughput.

Alternatively, the SSDs can be used. It uses the similar flash memory that is used in memory subsystem, thus increases the per-disk I/O throughput by almost 4-times than an HDD. Current SSDs offer almost 550MB/s I/O throughput in a moderate cost while maintaining considerable storage capacity. Due to the high bandwidth of SSDs, the most intuitive approach to build a high performance cluster is to use multiple SSDs with high end processors. However, the disk controllers are found to saturate at a certain point. So, adding large number of SSDs per node may incur huge cost without any performance gain.

3.0.3. Memory architecture

With the introduction of DDR (Double Data Rate) technology the memory bandwidth has been improved significantly because it reads one word of data during the positive edge and one word during the negative edge of the processor clock pulse. Most of the current HPC cluster as well as the computation clusters use either DDR2 or DDR3 RAM (random access memory or, simply main memory) modules. There is hardly any difference among current clusters (including traditional HPC clusters and datacenters) in terms of memory architecture or processor-memory communication model.

However, the total memory capacity of the cluster make significant difference in performance. That is, for improved performance, sufficient memory should be provided to the cluster that can hold the required computation in conjunction with the big data. Furthermore, more memory improves the caching effect which again improves the applications' performance. However, the main memory is costly which makes the job of building the balanced system complicated. Again, the SSD is nowadays increasingly considered as a new layer between memory (Since they use similar flash arrays as RAM) and storage due to its increased I/O throughput which may change the designer's approcah towards a balanced system in near future.

4. Cluster for Data- and Compute-intensive Genomic Application Amdahl's Second Law

From the above discussion it is clear that the system performance is constrained by its slowest component. We also discuss about the current state of design alternatives individually for network, storage and memory subsystem individually that the system designers should consider as they strive for a balanced system.

Computer scientist Gene Amdahl postulated several design principals in late 1960 for a balanced system. These design principals are collectively known as Amdahl's second law which are as follows:

1. Amdahl's I/O Law: A balanced computer system needs one bit of sequential I/O per second per instruction per second. This law is known as Amdahl's I/O number and can be expressed as Equation-1

$$Amdahl's\ I/O\ Number = \frac{I/O\ Bandwidth\ (bit/s)}{CPU\ Speed\ (GIPS)} \quad (1)$$

2. Amdahl's memory Law: A balanced computer system needs one byte of memory per instruction per second. It is known as Amdahl's memory number and can be expressed as Equation-2

$$Amdahl's\ Memory\ Number = \frac{Memory\ size\ (GB)}{CPU\ Speed\ (GIPS)} \quad (2)$$

From the perspective of genome scientists, it is clear that the future analytic systems will definitely use the multicore processors to meet scientists' dire need for speed and performance. However, several issues are yet to be clarified from the system designers' perspective. First, how the processing speed impacts the memory and I/O subsystem. In particular the system is constrained by its slowest component i.e the I/O subsystem. Hence, CPUs that can process data faster compared to its I/O subsystem can be counterproductive. Second, the price component of the compute cluster complicates the system designers' effort further. A system with huge I/O bandwidth and memory capacity with slower CPU will show lower performance per dollar. The situation is more serious in genomic research than other scientific areas because the falling cost of raw sequencing already outpaced Moore's Law and the corresponding linear increase in the storage and computation cost is about to hit the prohibitive wall. Hence, the question that is becoming increasingly important in both scientific as well as industrial community, *how does a next generation balanced computation cluster should look like which can handle the ever growing sequencing data efficiently yet in a cost effective way?*

5. Experimental Testbeds

5.1. Overview

As mentioned earlier, in our study we evaluate three different cluster architecture. Table-2 shows the overview of our experimental testbeds. The first one, SuperMikeII represents a traditional supercomputing cluster. This LSU HPC cluster offers a total 440 computing nodes. However, a maximum 128 can be allocated at a time to a single user. SwatIII represents a regular datacenter. This Samsung datacenter has 128 nodes. However, we use maximum 16 nodes for our experiments. The last one, CeresII is a novel hyperscale[8] system based on Samsung MicroBrick. We use as the baseline configuration and compare the performance of other clusters with respect to this. SwatIII is used as the main testbesd for our experiments. We change its configuration to study the pros and cons of different cluster architecture. It is to be noted, we always use homogeneous configuration across any cluster. We reported the performance and the price of different clusters in terms of the Hadoop datanodes (DN) only. For the masternode, we always use a minimal configuration based upon the resources in the cluster. In the subsequent sections we use the term node and datanode interchangeably.

5.2. Cluster Characterization with Amdahl's Second Law

In this section we characterize each cluster on the basis of Amdahl's I/O and memory number and discuss the major pros and cons of each different cluster architecture with respect to Amdahl's second Law. Since all the clusters in our evaluation use the same processor family (i.e., Intel Xeon) we use Equation-**??** and **??** to calculate the Amdahl's I/O and memory numbers of different clusters.

[8]"the ability of an architecture to scale appropriately as increased demand is added" (Source: https://en.wikipedia.org/wiki/Hyperscale)

5.2.1. Compute-centric architecture

SuperMikeII, being optimized for compute-intensive applications use two 8-core Intel Sandybridge Xeon processor per node thus offering huge processing power. However, each SuperMikeII node is equipped with only one HDD, thus, limited in terms of sequential I/O bandwidth. Also, each SuperMikeII node has only 32GB memory. It has the lowest Amdahl's number (both I/O and memory) among all the cluster and expected to suffer from I/O wait thus hindering real time analysis of any large scale genomic application.

SwatIII-Basic-HDD is similar in every aspect to SuperMikeII except it uses 10-Gbps Ethernet instead of 40-Gbps InfiniBand (with 2:1 blocking) as in SuperMikeII. We discuss the impact of network in details in Section-**??**

SwatIII-Basic-SSD, as the name suggests, is storage optimized and uses one SSD per node instead of one HDD as in SuperMikeII and SwatIII-Basic-HDD. Thus its sequential I/O throughput is increased comparing to the other two with a corresponding increase in Amdahl's I/O number. However, due to the high number of cores per node it is much less than the unity. Although the use of SSD alleviated the problem of HDD, the major bottleneck of this architecture is the lower amount storage and memory space which is very crucial in the context of high volume of next generation sequencing (NGS) data.

5.2.2. Data-centric scaled up architecture

Unlike SuperMikeII or SwatIII-Basic-HDD and -SSD, which use only one direct attached storage (DAS) device per node, SwatIII-FullScaleup-HDD and -SSD use 7 DAS per node using JBOD (Just a Bunch of Disk) configuration while using the same high end processor (i.e. two 8-core Intel SandyBridge Xeon). This architecture is called the scaled up architecture in conventional terminology. Since the sequential I/O throughput increases linearly with number of disks, both SwatIII-SSD and -HDD, have higher Amdahl's I/O number. Furthermore, teh SSD variant has higher number than the HDD variant. Regarding memory, each scaled up node has 256GB memory thereby, achieve high Amdahl's memory number also.

This architecture succesfully alleviates the major bottlenecks of the compute centric architecture both in terms of storage and memory space as well as I/O thorughput. However, the current high end disk controllers saturate after a certain amount of I/O throughput (an observation by Szalay [23]). In our Hadoop based assembly application we found this saturation point to arrive at 550MB/s. Hence, in this architecture we do not expect any advantage from SSD in terms of performance.

5.2.3. Data-centric scaled out architecture

The last one, CeresII is a Samsung MicroBrick-based scaled out architecture which uses a separate low power CPU for each SSD. CeresII has the highest Amdahl's I/O number (very close to unity) among all the clusters as well as appreciably high Amdahl's memory number, which reflect the proper balance between the processor, storage, and the memory subsystem. Each MicroBrick (or, simply the computation module) of CeresII consists of a 2-core Intel Xeon E3-1220L V2 processor, one

SATA-SSD (Samsung), and 16GB RAM. Each module has several PCI-express (PCIe) ports. Unlike SuperMikeII (traditional supercomputer) and SwatIII (regular datacenter), all the compute servers of CeresII in a single chassis are connected to a common PCIe switch to communicate with each other. The highly dense servers per chassis in CeresII have a total 44 physical cores connected through PCIe comparing to 16 physical cores per node as in SuperMikeII and SwatIII variants.

6. Workload

De novo genome assembly refers to the construction of an entire genome sequence from a huge amount of small, overlapping and erroneous fragments called short read sequences while no reference genome is available. The problem can be mapped as a simplified de Bruijn graph traversal [**?**]. We classified the de novo assembly in two stages as follows: 1) Hadoop-based de Bruijn graph-construction and 2) Giraph-based graph-simplification. In this section, we provide a brief overview of each stage of the assembler.

6.1. Hadoop-based De Bruijn graph-construction (data- and compute-intensive workload)

After filtering the actual short reads (i.e., the line containing only nucleotide characters *A*, *T*, *G*, and *C*) from a standard fastq file, an extremely shuffle-intensive Hadoop job creates the graph from these reads. the Hadoop map task divides each read into several short fragments of length *k* known as *k*-mers. Two subsequent *k*-mers are emitted as an intermediate key-value pair that represents a vertex and an edge (emitted from that vertex) in the de Bruijn graph. The reduce function aggregates the edges (i.e the value-list) of each vertex (i.e., the *k*-mer emitted as key) and, finally, writes the graph structure in the HDFS in the adjacency-list format. Based upon the value of *k* (determined by biological characteristics of the species), the job produces huge amount of shuffled data. For example, for a read-length of 100 and *k* of 31 the shuffled data size is found to be 20-times than the original fastq input. On the other hand, based upon the number of unique *k*-mers, the final output (i.e., the graph) can vary from 1 to 10 times of the size of the input.

6.1.1. Giraph-based Graph Simplification (memory- and compute-intensive workload)

The large scale graph data structure produced by the last MapReduce stage is analyzed here. This stage consists of a series of memory-intensive Giraph jobs. Each Giraph job consists of three different types of computation: compress linear chains of vertices followed by removing the tip-structure and then the bubble-structure (introduced due to sequencing errors) in the graph. Giraph can maintain a counter on the number of supersteps and the master-vertex class invokes each type of computation based on that. We compress the linear chains into a single vertex using a randomized parallel list ranking algorithm of [26] implented with Giraph. The computation proceeds in rounds of two supersteps until a user defined *limit* is reached. In one superstep, each compressible vertex with only one incoming and outgoing edge is labeled with either *head* or *tail* randomly with equal probability and send a message containing the tag to the immediate predecessor. In the next superstep, all the *head-tail* links are merged, that is, the *head-k*mer is extended (or, appended) with the last character of the *tail-k*mer and the *tail* vertex is removed. Each vertex also maintain a frequency counter which increments after each extension to that vertex. After the compression, all the tip-structures in the graph are removed in two supersteps. The first superstep identifies all the vertices with very short length (less than a threshold) and no outgoing edge as tips which are removed in the second superstep. Finally, the bubbles-structures are resolved in another two supersteps. In the first superstep, the vertices with same predecessor and successor as well as very short length (less than a threshold) are identified as bubbles. They send a message containing their id, value, and frequency to their corresponding predecessors. The predecessor employs a Levenshtein-like edit distance algorithm. If the vertices are found similar enough, then the lower frequency vertex is removed.

6.2. Input Data

High throughput next generation DNA sequencing machines (e.g., Illumina Genome Analyzer) produce a huge amount of short read sequences typically in the scale of several Gigabytes to Terabytes. Furthermore, the size of the de Bruijn graph built from these vast amount of short reads may be a magnitude higher than the reads itself making the entire assembly pipe line severely data-intensive.

7. Result and Analysis

7.1. Individual impact of different hardware components

In this section, we compare the impact of hardware components, such as, network, storage, and memory individually on our benchmark genome assembler. To do that, we use 16 nodes in both SuperMikeII and SwatIII. Each node in both the clusters has 16 processing cores. We started with comparing the impact of the network between SuperMikeII and SwatIII-Basic-HDD. Then, we further optimized those 16 nodes of SwatIII cluster incrementally in terms of storage by providing SSD (named as SwatIII-Basic-SSD) and then providing more memory to each node (named as SwatIII-Memory). It is worthy to mention here that 1-HDD/node puts a practical limit on the number of concurrently running Yarn-containers due to I/O bottleneck. As a consequence, both in SuperMikeII and SwatIII-Basic-HDD, we observed the best performance by using only 8 Yarn-containers concurrently in each node, i.e., only half of the number of cores per node. The execution-times reported in this section are the means of at least 3 runs of the assembler on each different hardware configuration.

7.1.1. Effect of network architecture: Fat tree InfiniBand vs Clos fabric Ethernet

Figure-1a compares the impact of network interconnect on PGA's genome assembly pipeline while assembling a 90GB bumble bee genome. The execution time is normalized to the SuperMikeII-baseline. We did not find any visible performance

difference (less than 2%) on any of the stages of our assembly pipeline even though SuperMikeII uses 40-Gbps QDR InfiniBand whereas SwatIII-Basic-HDD uses a 10-Gbps ethernet. although the average latency in SuperMikeII is almost $1/14$ of that in SwatIII (0.014ms in SuperMikeII compare to 0.2ms in SwatIII), the average effective bandwidth between any two compute nodes of SuperMikeII was found to be almost 10-times lower than that of SwatIII (954Mbit/s in SuperMikeII, whereas 9.2Gbit/s in SwatIII) because of the $2:1$ blocking ratio in the InfiniBand network. Considering the network architecture, SwatIII-Basic although uses Clos fabric architecture shows similar performance because of its very low Amdahl's numbers.

7.1.2. Effect of storage architecture: HDD and SSD

Figure-1b compares the execution time of SwatIII-Basic-SSD (1-SSD/node) to the SuperMikeII-baseline (1-HDD/node). The second column of each stage of the assembler in Figure-1b shows the impact of using SSD in that stage of the assembly. We observed almost 50% improvement in the shuffle intensive graph-construction stage because of reduced I/O wait. However, graph-simplification, a series of in-memory Giraph jobs (that read/write data only to the HDFS), is not affected much (less than 3%) by storage optimization with SSD. Actually, the shuffle phase of the Hadoop job experiences a huge amount of I/O wait.

Figure-1c on the other hand, compares the impact of multiple storage devices (scaled up) per node. We temporarily replace the SSD of the SwatIII-basic-SSD cluster and added HDDs until the disk controller saturates its limit. The performance is found to improve almost linearly with addition of HDD, thus increasing the sequential I/O throughput. For the Hadoop job, the saturation is found to reach at 4-HDDs per node, i.e., a throughput of 600MB/s. On the other hand, 4-HDDs per node shows similar performance (only 5% variation) with a single-SSD (i.e., an I/O throughput of 520MB/s) per node. Once the disk controller is saturated, adding more disk(s) to the datanodes does not improve the performance.

7.1.3. Effect of size of DRAM

To observe the impact of size of DRAM, we increased the memory of each node of the SwatIII -Basic-SSD cluster from 32GB to 256GB temporarily thus increasing the Amdahl's memory number. The third columns of Figure-1b shows the impact of increasing the amount of memory per node. We observed almost 20% improvement in the initial graph-construction phase from SwatIII-Basic-SSD, consequently, almost 70% improvement to the baseline. In the Giraph phase, the corresponding improvement is almost 35%. The improvement is because of the caching especially in case of Giraph, where computation proceeds in iterative supersteps. A huge amount of data is kept in cache and is fetched upon requirement during the next compute-superstep.

7.2. Performance of Different Cluster Architecture

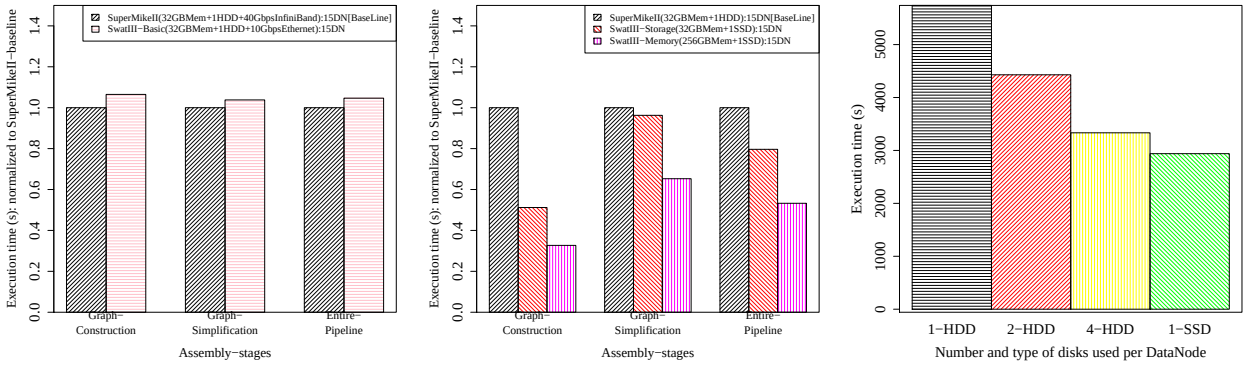
7.2.1. Relative merits of each cluster with respect to fixed Hardware Investment cost

Table-3 compares the above architectures while keeping the total cost same as that of the number of nodes used in SuperMikeII (i.e., 15). Table-1 shows the cost of each individual hardware component that are used in our testbeds. The explanation of each column of Table-3 is as follows:

1. Total Cost (say c) = The cost of 15 Nodes of SuperMikeII (i.e. \$57060) is considered as baseline and kept constant for all clusters.
2. Cost/DN (say c_d) is calculated with current maket price of processor, SSD/HDD, and memory
3. $\#DN = c/c_d$ (it shows the #DN available in that constant cost)
4. Total Processing Speed = $\#DN \times (\#cores/DN) \times CoreSpeed$
5. Total Storage = $\#DN \times (\#Disk/DN) \times SpacePerDisk$
6. Total Memory = $\#DN \times Memory/DN$
7. Projected Execution Time is calculated on the basis of our previous work assuming a linear scalability for any cluster.

Following are the observations

1. **SuperMikeII**, the traditional HPC cluster has the highest processing power among all. However it is limited in terms of sequential I/O throughput as well as total storage and memory space.
2. **SwatIII-basic-SSD** offers similar processing power as of SuperMikeII. It also provides significantly higher I/O bandwidth. Thus it is expected to perform fastest among all the clusters. However, it is severely limited in terms of total storage and memory space.
3. **SwatIII-FullScaleup-HDD** offers the maximum amount of storage and memory sapce. Apperently it seems to be the most optimal architecture in terms of processing power also.
4. **SwatIII-FullScaleup-SSD** has the same characteristics as of SwatIII-FullScaleup-HDD. But due to the significantly higher cost of SSD than HDD, it offers lower processing power, total storage and memory space than its HDD variant. Hence, for a scaled up architecture, it is always recommended to use multiple HDD rather than SSD.
5. **CeresII**, the MicroBrick based architecture is the most balanced which offers a similar processing speed compared to the scaled up archtectures. Although it offers lower storage and memory space than that of the scaled up cases the dececreasing cost of SSD will soon overcome the storage bottleneck. Figure-**??** shows the performance of different cluster architecture while assembling the bumble bee genome keeping the total cost of the cluster same.



(a) Effect of Network: Fat tree 40Gbps InfiniBand vs Clos fabric 10Gbps Ethernet

(b) Effect of storage (HDD and SSD) and DRAM size

(c) Performance trend with multiple HDDs. 4-HDDs or 1-SSD saturates disk controller

Figure 1: Performance trend using HDD and SSD in Hadoop

6. **Questions I got after comparing SwatIII-FullScaleup-HDD and CeresII:** [Q1] Why CeresII a better architecture than SwatIII-FullScaleup-HDD? As it can be seen, In the same cost, SwatIII-FullScaleup-HDD provides more processing power, more storage space (almost double), more memory space (again, almost double) than CeresII. Is it because the total cost of ownership including power dissipation and space consumption? Can we get these data? [Q2] Although CeresII is projected to perform better for Hadoop workload while keeping the cost same, it is actually contradictory to the above theoratical cost and performance analysis (In both the cases there is no I/O wait). Is it because of PCIe vs Ethernet? Is it possible to get the real data for SwatIII-FullScaleup-HDD(8DN) and CeresII(65DN) as shown in the table?

7.2.2. Performance of Different Cluster Architectures keeping the same Hardware Investment Cost

7.3. Total Cost of Ownership

8. Tables

9. Reference

[1] M. Massie, F. Nothaft, C. Hartl, C. Kozanitis, A. Schumacher, A. D. Joseph, D. A. Patterson, Adam: Genomics formats and processing patterns for cloud scale computing, EECS Department, University of California, Berkeley, Tech. Rep. UCB/EECS-2013-207.

[2] J. Gray, P. Shenoy, Rules of thumb in data engineering, in: Data Engineering, 2000. Proceedings. 16th International Conference on, IEEE, 2000, pp. 3–10.

[3] A. S. Szalay, J. A. Blakeley, Gray's laws: database-centric computing in science. (2009).

[4] R. Appuswamy, C. Gkantsidis, D. Narayanan, O. Hodson, A. Rowstron, Scale-up vs scale-out for hadoop: Time to rethink?, in: Proceedings of the 4th annual Symposium on Cloud Computing, ACM, 2013, p. 20.

[5] M. Michael, J. E. Moreira, D. Shiloach, R. W. Wisniewski, Scale-up x scale-out: A case study using nutch/lucene, in: Parallel and Distributed Processing Symposium, 2007. IPDPS 2007. IEEE International, IEEE, 2007, pp. 1–8.

[6] L. D. Stein, et al., The case for cloud computing in genome informatics, Genome Biol 11 (5) (2010) 207.

[7] M. Schatz, The next 20 years of genome research, bioRxiv (2015) 020289.

Hardware component	**Used in**	**Cost ($)**
Intel SandyBridge Xeon 64bit Ep series (8-cores) processor	SuperMikeII, SwatIII	1766
Intel Xeon E3-1220L V2 (2-cores) processor	CeresII	384
Western Digital RE4 HDD	SuperMikeII	132
Western Digital VelociRaptor HDD, 500GB	SwatIII HDD-variants	157
Samsung 840Pro Series SATAIII SSD, 500GB	SwatIII SSD-variants	450
Samsung 840Pro Series SATAIII SSD, 250GB	CeresII	258
Samsung DDR3 16GB memory module	SwatIII, CeresII	159
32GB 1600MHz RAM (decided by Dell)	SuperMikeII	140 (Average)

Table 1: Hardware components used in different cluster configurations, and their cost

	Job Type	Input	Final output	# jobs	Shuffled data	HDFS Data
Graph Construction	Hadoop	90GB (500-million reads)	95GB	2	2TB	136GB
Graph Simplification	Series of Giraph jobs	95GB (71581898 vertices)	640MB (62158 vertices)	15	-	966GB

Table 4: Moderate-size bumble bee genome assembly

Cluster	cores / DN [a]	Speed / core [b]	Total cpu speed (GHz) [c=a*b]	Space / Disk (GB) [d]	Seq IO BW / Disk (MB/s) [e]	#Disk / DN [f]	Total Seq IO BW (GB/s) [g=e*f]	Memory / DN (GB)[h]	Amdahl IO No [i=g/c*8]	Amdahl Mem No [j=h/c]	Cost / DN ($) [k]
SuperMikeII (Traditional Supercomputer) [BaseLine]	16	2.6	41.6	500	150	1-HDD	0.146	32	0.028	0.769	3804
SwatIII-Basic-HDD (Regular Datacenter)	16	2.6	41.6	500	150	1-HDD	0.146	32	0.028	0.769	3829
SwatIII-Basic-SSD (Regular Datacenter)	16	2.6	41.6	500	520	1-SSD	0.507	32	0.097	0.769	4300
SwatIII-FullScaleup-HDD (Regular Datacenter)	16	2.6	41.6	500	150	7-HDD	1.025	256	0.197	6.153	6390
SwatIII-FullScaleup-SSD (Regular Datacenter)	16	2.6	41.6	500	520	7-SSD	3.554	256	0.683	6.153	9226
CeresII (Hyperscale System)	2	2.3	4.6	250	520	1-SSD	0.507	16	0.883	3.478	879

Table 2: Experimental testbeds with different configuration

Cluster	Total Cost ($)	Cost/DN($)	#DN	Total Processing Speed (GHz)	Total Storage (GB)	Total Memory (GB)	Projected Exec Time (s) for Hadoop
SuperMikeII (32GBMem + 1HDD) : 15DN [BaseLine]	57060	3804	15	624	7500	480	5740
SwatIII-Basic-HDD (32GBMem + 1HDD)	57060	3829	15	619.92	7451.03	476.86	6122
SwatIII-Basic-SSD (32GBMem + 1SSD)	57060	4300	13	552.02	6634.88	424.63	3335
SwatIII-FullScaleup-HDD (256GBMemory + 7HDD)	57060	7175	8	330.83	27832.32	2037.76	3162
SwatIII-FullScaleup-SSD (256GBMemory + 7SSD)	57060	9226	6	257.28	21646.43	1583.28	3330
CeresII	57060	879	65	298.60	16228.66	1038.63	2583

Table 3: Configuration available for different architecture keeping the cost constant

	Job Type	Input	Final output	# jobs	Shuffled data	HDFS Data
Graph Construction	Hadoop	452GB (2-billion reads)	3TB	2	9.9TB	3.2TB
Graph Simplification	Series of Giraph jobs	3.2TB (1483246722 vertices)	3.8GB (23032297 vertices)	15	-	4.1TB

Table 5: Large-size human genome assembly

[8] K. Bhuvaneshwar, D. Sulakhe, R. Gauba, A. Rodriguez, R. Madduri, U. Dave, L. Lacinski, I. Foster, Y. Gusev, S. Madhavan, A case study for cloud based high throughput analysis of ngs data using the globus genomics system, Computational and Structural Biotechnology Journal 13 (2015) 64–74.

[9] N. Siva, 1000 genomes project, Nature biotechnology 26 (3) (2008) 256–256.

[10] B. Langmead, M. C. Schatz, J. Lin, M. Pop, S. L. Salzberg, Searching for snps with cloud computing, Genome Biol 10 (11) (2009) R134.

[11] T. White, Hadoop: the definitive guide: the definitive guide, " O'Reilly Media, Inc.", 2009.

[12] H. Nordberg, K. Bhatia, K. Wang, Z. Wang, Biopig: a hadoop-based analytic toolkit for large-scale sequence data, Bioinformatics (2013) btt528.

[13] C. Olston, B. Reed, U. Srivastava, R. Kumar, A. Tomkins, Pig latin: a not-so-foreign language for data processing, in: Proceedings of the 2008 ACM SIGMOD international conference on Management of data, ACM, 2008, pp. 1099–1110.

[14] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, I. Stoica, Spark: cluster computing with working sets, in: Proceedings of the 2nd USENIX conference on Hot topics in cloud computing, Vol. 10, 2010, p. 10.

[15] Y. Kang, Y.-s. Kee, E. L. Miller, C. Park, Enabling cost-effective data processing with smart ssd, in: Mass Storage Systems and Technologies (MSST), 2013 IEEE 29th Symposium on, IEEE, 2013, pp. 1–12.

[16] D. Wu, W. Luo, W. Xie, X. Ji, J. He, D. Wu, Understanding the impacts of solid-state storage on the hadoop performance, in: Advanced Cloud and Big Data (CBD), 2013 International Conference on, IEEE, 2013, pp. 125–130.

[17] S. Moon, J. Lee, Y. S. Kee, Introducing ssds to the hadoop mapreduce framework, in: Cloud Computing (CLOUD), 2014 IEEE 7th International Conference on, IEEE, 2014, pp. 272–279.

[18] K. Krish, A. Khasymski, G. Wang, A. R. Butt, G. Makkar, On the use of shared storage in shared-nothing environments, in: Big Data, 2013 IEEE International Conference on, IEEE, 2013, pp. 313–318.

[19] B. Li, E. Mazur, Y. Diao, A. McGregor, P. Shenoy, A platform for scalable one-pass analytics using mapreduce, in: Proceedings of the 2011 ACM SIGMOD International Conference on Management of data, ACM, 2011, pp. 985–996.

[20] W. Tan, L. Fong, Y. Liu, Effectiveness assessment of solid-state drive used in big data services, in: Web Services (ICWS), 2014 IEEE International Conference on, IEEE, 2014, pp. 393–400.

[21] G. Bell, J. Gray, A. Szalay, Petascale computational systems, Computer 39 (1) (2006) 110–112.

[22] A. S. Szalay, G. Bell, J. Vandenberg, A. Wonders, R. Burns, D. Fay, J. Heasley, T. Hey, M. Nieto-SantiSteban, A. Thakar, et al., Graywulf: Scalable clustered architecture for data intensive computing, in: System Sciences, 2009. HICSS'09. 42nd Hawaii International Conference on, IEEE, 2009, pp. 1–10.

[23] A. S. Szalay, G. C. Bell, H. H. Huang, A. Terzis, A. White, Low-power amdahl-balanced blades for data intensive computing, ACM SIGOPS Operating Systems Review 44 (1) (2010) 71–75.

[24] D. Cohen, F. Petrini, M. D. Day, M. Ben-Yehuda, S. W. Hunter, U. Cummings, Applying amdahl's other law to the data center, IBM Journal of Research and Development 53 (5) (2009) 5–1.

[25] F. Liang, C. Feng, X. Lu, Z. Xu, Performance characterization of hadoop and data mpi based on amdahl's second law, in: Networking, Architecture, and Storage (NAS), 2014 9th IEEE International Conference on, IEEE, 2014, pp. 207–215.

[26] U. Vishkin, Randomized speed-ups in parallel computation, in: Proceedings of the sixteenth annual ACM symposium on Theory of computing, ACM, 1984, pp. 230–239.