International Journal of Advance Research in Science and Engineering
Volume No.06, Special Issue No.(04), December 2017
www.ijarse.com
IJARSE
ISSN (O) 2319 - 8354

cHigh Performance Computing:

Initial Boost, Exploitation and Application

Neel Patel[1], Neil Patel[2], Manan Doshi[3], Yash Shah[2]

[1,2,3,4]*Department of Computer Engineering, Shri Bhagubhai Mafatlal Polytechnic (India)*

ABSTRACT

Computational tasks of high-end enterprises, research centre, educational institutes are demanding for huge data storage capacity and better throughput. These crucial issues were resolved by the introduction of a concept, called high performance computing. The development of supercomputers, which incorporated parallel processing techniques, helped realize this concept. This paper explains the realization of high performance computing through the development of parallel processing and supercomputers over the years. It also enlists the compute-intensive applications which employ high performance computing to deliver reliable, accurate and quick results.

Keywords: Cluster, MPI, Share-everything architecture, Share- nothing architecture

I. INTRODUCTION

In a world of digitization, and machine-learning, every piece of information around us, ranging from the current temperature to passwords and pin numbers, status updates, photos, selfies, snapchat stories, everything, gets stored as data in databases. When we here the term "databases", what usually comes to our mind is voluminous data and in today's era, largely voluminous. With the rise of data volumes, worldwide, there was much need of newer storage techniques, and processing of this huge data.

The increase in complex simulations, 3-D modeling, genome structuring and such bio-chemical or meteorological applications, and graphical applications, has led to an urge to consume more processor speed, and quicker response-time.

One single commodity desktop computer wouldn't be able to complete these high-computational workloads, or would require months or even years, to complete.

With emergence of technologies like parallel processing and creation of high-end computers, referred to as supercomputers or exotic machines, a new type of computing evolved. This new computing is known as High Performance Computing. This paper talks about the evolution of high performance computing, its types and working.

II. HIGH PERFORMANCE COMPUTING

High-performance computing (HPC) is the use of parallel processing for running advanced application programs efficiently, reliably and quickly. The term applies especially to systems that function above a teraflop or 1012 floating-point operations per second. The term HPC is occasionally used as a synonym for supercomputing, lthough technically a supercomputer is a system that performs at or near the currently highest

operational rate for computers. Some supercomputers work at more than 1015 floating-point operations per second (petaflop). [1] HPC brings together several technologies such as computer architecture, algorithms, electronics, system softwares, specialized operating systems and application programs under a single canopy to solve advanced problems effectively and quickly. A highly efficient HPC system requires an InfiniBand (IB) to connect multiple nodes and clusters.

[2] [3]

III. INITIAL BOOST TO HIGH PERFORMANCE COMPUTING

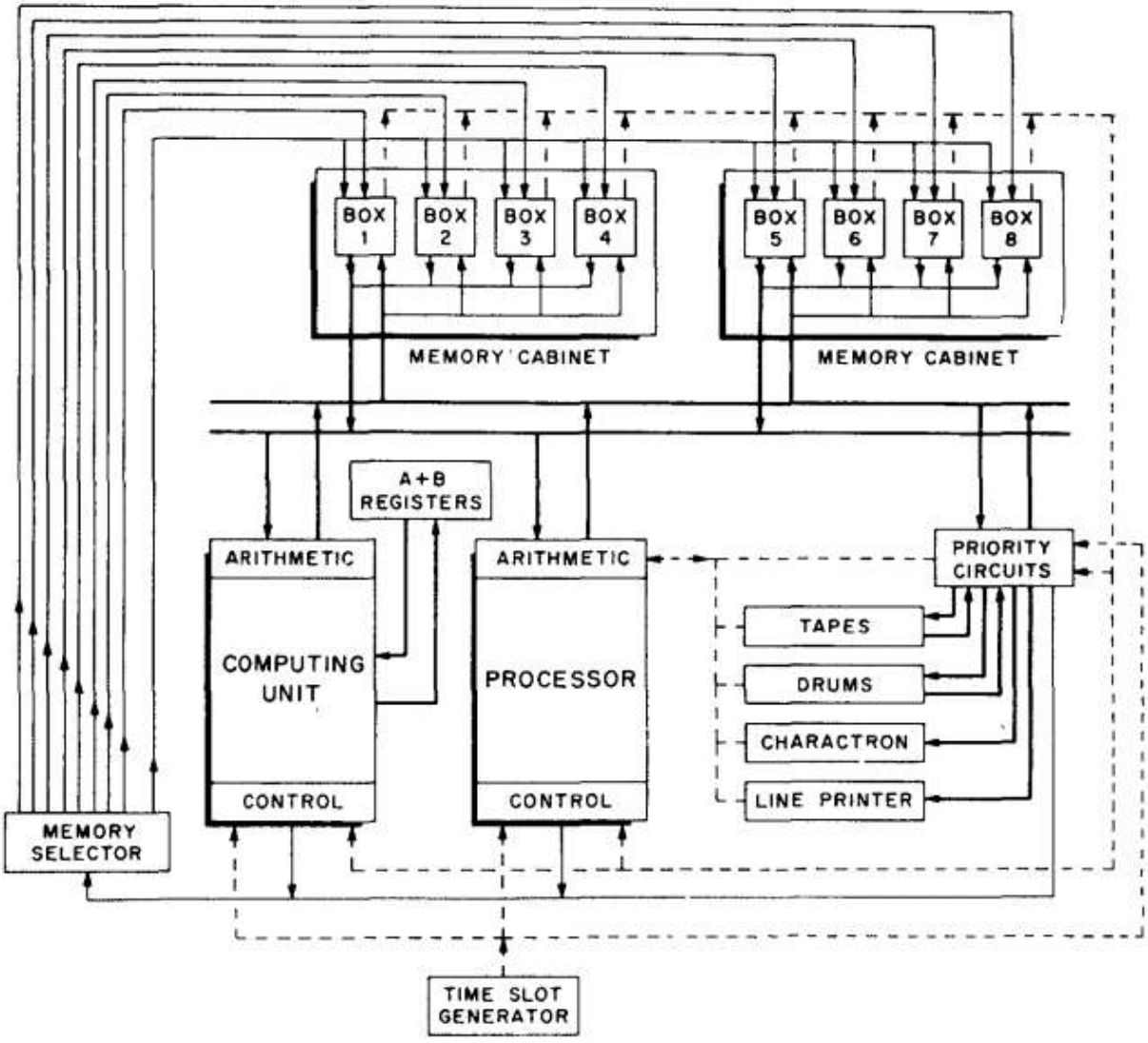

Figure 1 – LARC Architecture [4]

1. It was massive both in physical size and computational capability. A new building (now numbered B-117) was constructed just to house it. Physically, it consisted of four major cabinets. Each cabinet was approximately twenty feet long, four feet wide, and seven feet tall. In addition there was an array of twelve floating-head

drums, each approximately four feet wide, three feet deep and five feet tall. There were eight tape units that used metallic tape (each tape weighed about ten pounds), a punched-card reader, and a large printer.

2. Operation of the system was controlled from a central console with lots of flashing lights, switches, digital readouts, and pushbuttons. Teletype units (with paper tape readers / punches) provided direct communications with the computer. The control console also provided an array of toggle switches to feed direct commands to the system.

3. The four hardware cabinets consisted of: the I/O processor unit (one cabinet) where all information destined to and from the drums, tapes, printers, console, etc. was routed and controlled; the computing unit (one cabinet, where computational activity occurred); and memory (two cabinets, each with 16K of ferrite core memory). Later, a third memory cabinet was added to expand local memory.

4. Arithmetic was performed in decimal mode, which was the custom at Univac at the time (the Univac I also performed decimal arithmetic). The LARC employed twelve decimal digits. A five-bit register represented the numerical value in each digit. Arithmetic was performed using these coded digits in a dizzying array of temporary storage registers that saved the initial integer values as well as partially computed results. There were storage registers, shift registers, and result registers that could store information for repetitive calculations that would ultimately yield an answer. Calculation speed was also dizzying: a 12 x 12 digit addition or subtraction could be accomplished in 4 microseconds, while a 12 x 12 multiplication would complete in twelve microseconds. Division

b. IBM 7030 Stretch

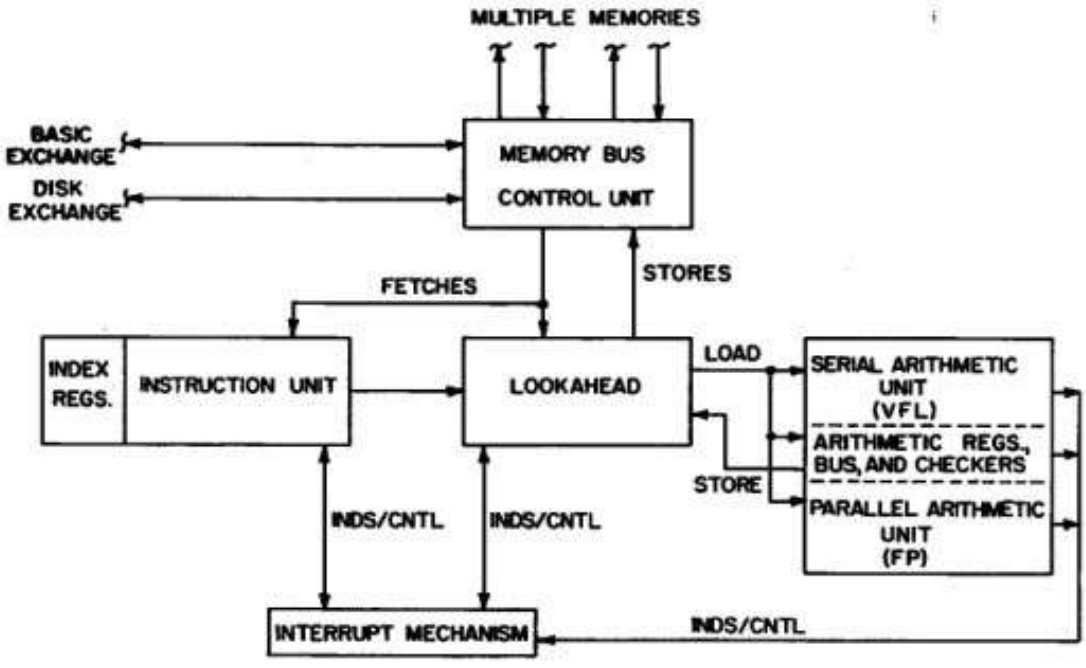

FIGURE 2. STRETCH COMPUTER

Figure 2 – Stretch Architecture [6]

1. Instructions in Stretch flowed through two processing elements: an indexing and instruction unit that fetched, predecoded, and partially executed the instruction stream, and an arithmetic unit that executed the remainder of the instructions.

2. Stretch also partitioned its registers according to this organization: a set of 16 64-bit index registers was associated with the indexing and instruction unit, and a set of 64-bit accumulators and other registers were associated with the arithmetic unit. The indexing and instruction unit of Stretch fetched 64-bit memory words into a two-word instruction buffer. Instructions could be either 32 or 64 bits in length, so up to four instructions could be buffered.

3. The indexing and instruction unit directly executed indexing instructions and prepared arithmetic instructions by calculating effective addresses (i.e., adding index register contents to address fields) and starting memory operand fetches. The unit itself was a pipelined computer, and it decoded instructions in parallel with execution. One interesting feature of the instruction fetch logic was the addition of predecoding bits to all instructions; this was done one word at a time, so two half-word instructions could be predecoded in parallel. [7]

4. The 7030 CPU used emitter-coupled logic (originally called current-steering logic) on 18 types of Standard Modular System (SMS) cards. It used 4,025 double cards and 18,747 single cards, holding 169,100 transistors, requiring a total of 21 kW power. It used high-speed NPN and PNP germanium drift transistors, with cut-off frequency over 100 MHz, and using ~50mW each. [8]

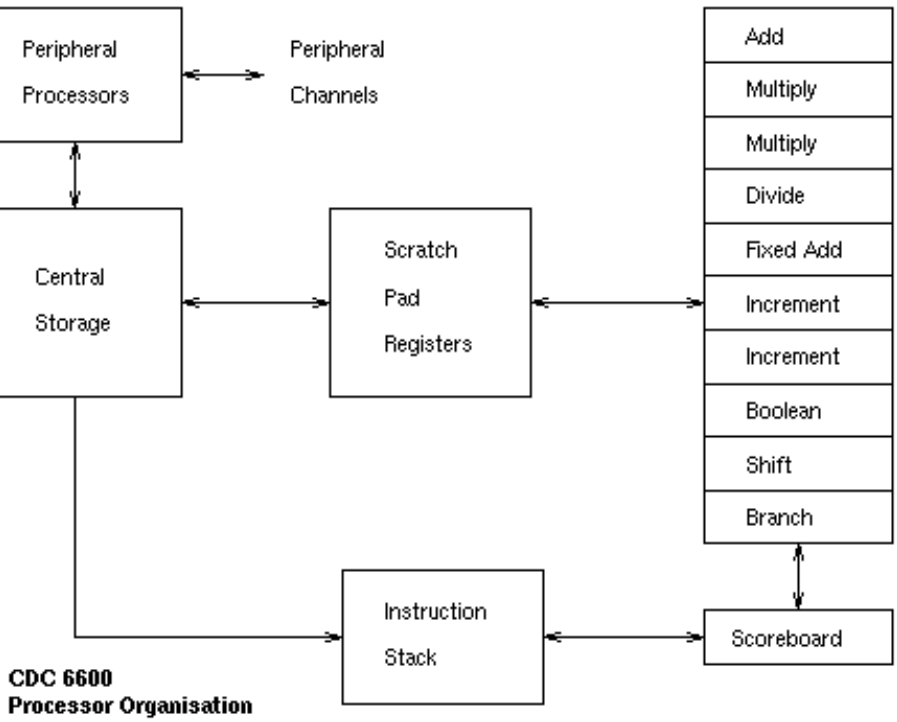

Figure 3 – CDC 6600 Architecture [9]

1. 10 independent "Functional Units" in the Main Processor included 2 floating point Multipliers (1 microsecond), 1 floating point Divider (3.4 microseconds), 1 Add and 1 Long Add units (0.3 microseconds), 2 increment (used for memory access), 1 branch, 1 Boolean, 1 shift unit (each 0.1 microseconds) and 1 population count (number of 1 bits in a word - go ask NSA why)

2. 10 (optionally 20) built in "Peripheral Processors" (PPs) controlled the main processor, operated the control console (including painting the alphanumerics in vector mode), and performed all I/O (Input/output)

3. Memory bandwidth was 1 60 bit word per 100 nanoseconds (10 per microsecond)

4. Memory access time was 475 nanoseconds

5. X shape helped reduce signal transit time. Outer parts of the X held not time critical things like heat exchangers, refrigerator pumps, "Dead Start" panel. [10]

6. In the 6600 Computer only silicon components are used. High density packaging in the 6600 is achieved through a unique packaging design. The technique employed results in more components per cubic inch. [11]

IV. TYPES OF HIGH PERFORMANCE COMPUTING

a. ***The commodity HPC cluster:*** Over the last ten years, the HPC cluster has disrupted the entire supercomputing market. Built from standard off-the-shelf servers and high speed interconnects, a typical HPC system can deliver industry-leading, cost-effective performance. A typical cluster can employ hundreds, thousands, and even tens of thousands of servers all working together on a single problem (this is the high tech equivalent of a "divide and conquer" approach to solving large problems). Because of high performance and low cost, the commodity cluster is by far the most popular form of HPC computing. Also keep in mind the compatibility advantage — x86 commodity servers are ubiquitous.

b. ***Dedicated supercomputer:*** In the past, the dedicated supercomputer was the only way to throw a large number of compute cycles at a problem. Supercomputers are still produced today and often use specialized non-commodity components. Depending on your needs, the supercomputer may be the best solution although it doesn't offer the commodity price advantage.

c. ***HPC cloud computing:*** This method is relatively new and employs the Internet as a basis for a cycles-as-a-service model of computing. The compute cycles in question live in the cloud somewhere allowing a user to request remote access to cycles on-demand. An HPC cloud provides dynamic and scalable resources (and possibly virtualization) to the end-user as a service. Although clouds can be cost effective and allow HPC to be purchased as an expense and not a capital asset, it also places some layers between the user and hardware that may reduce performance.

d. ***Grid computing:*** Grid is similar to cloud computing, but requires more control by the end-user. Its main use is academic projects where local HPC clusters are connected and shared on a national and international level. Some computational grids span the globe while others are located within a single organization. [12]

V. ARCHITECTURE

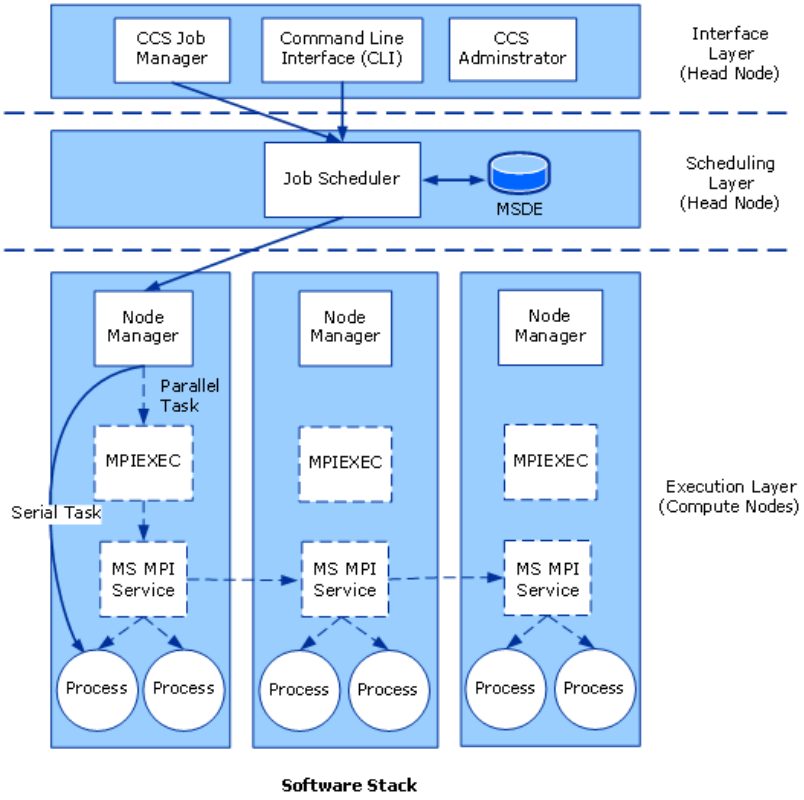

Figure 4 – HPC Software Stack with Microsoft's implementation of MPI [13]

The interface and scheduling layers reside on the head node. The execution layer resides primarily on the compute nodes. The execution layer as shown here includes the Microsoft implementation of MPI, called MS MPI, which was developed for Windows and is included in the Microsoft® Compute Cluster Pack.

a. Interface layer

The user interface layers consist of the Compute Cluster Job Manager, the Compute Cluster Administrator, and Command Line Interface (CLI).

The Compute Cluster Job Manager is a WIN32 graphic user interface to the Job Scheduler that is used for job creation and submission.

The Compute Cluster Administrator is a Microsoft Management Console (MMC) snap-in that is used for configuration and management of the cluster.

The Command Line Interface is a standard Windows command prompt which provides a command-line alternative to use of the Job Manager and the Administrator.

b. Scheduling layer

The scheduling layer consists of the Job Scheduler, which is responsible for queuing the jobs and tasks, reserving resources, and dispatching jobs to the compute nodes.

c. Execution layer

In this example, the execution layer consists of the following components replicated on each compute node: the Node Manager Service, the MS MPI launcher mpiexec, and the MS MPI Service.

The Node Manager is a service that runs on all compute nodes in the cluster. The Node Manager executes jobs on the node, sets task environmental variables, and sends a heartbeat (health check) signal to the Job Scheduler at specified intervals (the default interval is one minute).

The MS MPI Service is responsible for starting the job tasks on the various processors.

d. Task Flow

A task flow job is one in which a set of unlike tasks are executed in a prescribed order, usually because one task depends on the result of another task.

A task flow job is illustrated in the following figure. Note that only Tasks 2 and 3 are performed in parallel, because neither is dependent on the other. [13]

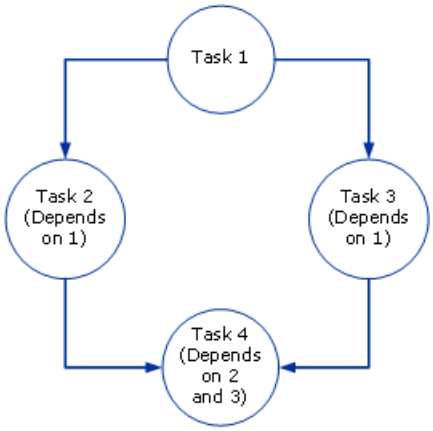

Figure 5 – Task Flow [13]

VI. APPLICATIONS OF HIGH PERFORMANCE COMPUTING

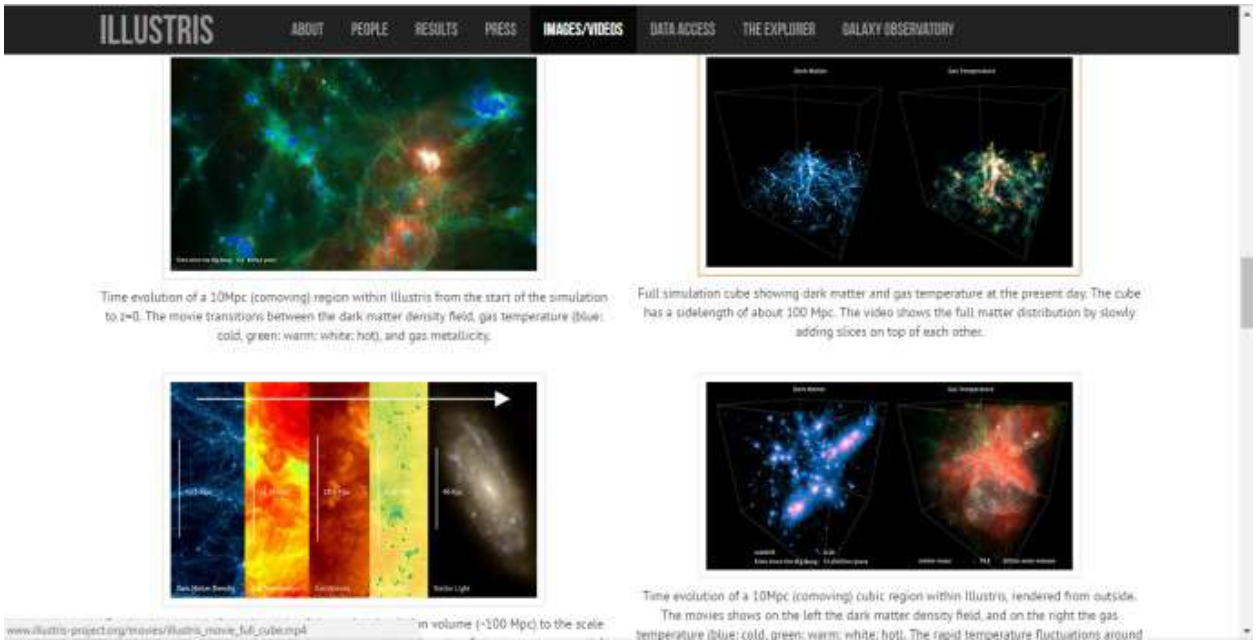

Figure 8 – Illustris – an application of HPC [18]

The Illustris project is a set of large-scale cosmological simulations, including the most ambitious simulation of galaxy formation yet performed. The calculation tracks the expansion of the universe, the gravitational pull of matter onto itself, the motion or "hydrodynamics" of cosmic gas, as well as the formation of stars and black holes. These physical components and processes are all modeled starting from initial conditions resembling the very young universe 300,000 years after the Big Bang and until the present day, spanning over 13.8 billion years of cosmic evolution. The simulated volume contains tens of thousands of galaxies captured in high-detail, covering a wide range of masses, rates of star formation, shapes, sizes, and with properties that agree well with the galaxy population observed in the real universe. [18]

VII. OPPORTUNITIES

a. **Speed Improvements**

As the amount of data grows, engineering has become core to every aspect of an organization. It's mission-critical to have systems with continuous uptime – real-time response is a necessity. "This is where the industry is moving: to crunch information very quickly, right inside computer memory, at speeds never heard before," says Sunder Singh (TATA Consultancy Services). Testing and iteration will yield to gradual improvements in speed that over time appear drastic.

b. **Information Management**

There is more data being collected now than ever before. Imagine every data point on the Internet. Now think about a universe where a limitless archive exists. One day, this information will be accessible, and HPC will be necessary to retrieve it. HPC will simplify the process of managing and crafting analysis from these archives. The technology will help ensure that every data point is part of a larger story.

c. **Immediate Access to Data**

There is always a lag time between information collection and analysis. HPC, however, is shortening this delay. In addition to increasing speeds in delivery, HPC will help organization process new data in real-time. **[19]**

VIII. CONCLUSION

The progress of high performance computing can be judged by decisions taken in the favour of high performance computing's development, research and utilization. Furthermore, technologies are being developed to make it ubiquitous and enable users to it's full exploitation. High Performance Computing's collaboration with Big Data and Computer Vision applications provides a best of both worlds approach to the increasing need of high computational power.

REFERENCES

[1] http://searchenterpriselinux.techtarget.com/definition/high-performance-computing
[2] https://www.techopedia.com/definition/4595/high-performance-computing-hpc
[3] https://en.wikipedia.org/wiki/InfiniBand
[4] https://www.computer.org/csdl/proceedings/afips/1956/5049/00/50490016.pdf
[5] http://www.computer-history.info/Page4.dir/pages/LARC.dir/LARC.Cole.html

[6] https://people.cs.clemson.edu/~mark/stretch/blosk_60.jpg

[7] https://people.cs.clemson.edu/~mark/stretch.html

[8] https://en.wikipedia.org/wiki/IBM_7030_Stretch

[9] http://homepages.inf.ed.ac.uk/rni/comp-arch/Paru/gifs/6600-cpu.gif

[10] http://ed-thelen.org/comp-hist/vs-cdc-6600.html

[11] http://archive.computerhistory.org/resources/text/CDC/CDC.6600.1963.102621029.pdf

[12] Douglas Eadline, HPC For Dummies

[13] https://technet.microsoft.com/en-us/library/cc720072(v=ws.10).aspx

[14] http://www.hector.ac.uk/cse/distributedcse/reports/cp2k03/cp2k03/images/dbcsr_cannon.png

[15] https://en.wikipedia.org/wiki/Cannon's_algorithm

[16] http://www.inf.fh-flensburg.de/lang/papers/trans/mesh.gif

[17] https://www.cs.auckland.ac.nz/~jmor159/363/html/systolic.html

[18] http://www.illustris-project.org/about/

[19] http://www.brightcomputing.com/blog/3-biggest-opportunities-in-hpc