Parallel Computing – Different Approaches

Poonkodi Ponnambalam
Computer Science Department
San Jose State University
San Jose, CA 95192
408-924-1000
poongodee@gmail.com

ABSTRACT

One of the key contributors to the explosion of the information and Internet age is the ability to parallelize computation. The theoretical limits of computation with a single machine are long superseded by massively scalable parallel computers. This paper discusses about the multiple different levels where parallelism can be achieved. Two of the most game-changing technologies are discussed here – namely, the SuperComputers and MapReduce. We also discuss one of the most important applications of today's Big Data technology – Graph partition.

1. INTRODUCTION

Parallelism can be achieved in multiple levels and this paper explains how problems can be parallelized in both hardware and software level. At the hardware level, parallelism can be achieved by replicating the number of computation units in the system. One of the famous examples is the IBM Supercomputer BlueGene/L which is a massively parallel computer with 65,536 nodes and system-on-a-chip technology with processing power of 360 teraFLOPS (trillion floating-point operations per second). In the software paradigm, MapReduce has changed the way we process very large amount of data. It is a programming framework to process large volume of data on a cluster using parallel and distributed algorithms.

2. MOTIVATION

Parallel computation has several different applications from Search Engines, Social networking sites, Internet Commerce. Companies like Google, Facebook, Amazon, Microsoft would not exist without the advances in parallel computing. Google serves over 100 million queries every single day, with latencies of less than 100 milli seconds. Graph partitioning is a problem that is central to the Internet. We discuss the partitioning problem in detail and how parallel computation has revolutionized it.

3. PARALLEL COMPUTING IN HARDWARE

Traditionally, parallel computing was done in hardware. The number of microprocessors in the hardware multiplied with the advent of transistor technologies and enabled having multiple computation units within the system. Moore's law guided the technologists for several decades doubling the transistor count every 18 months.

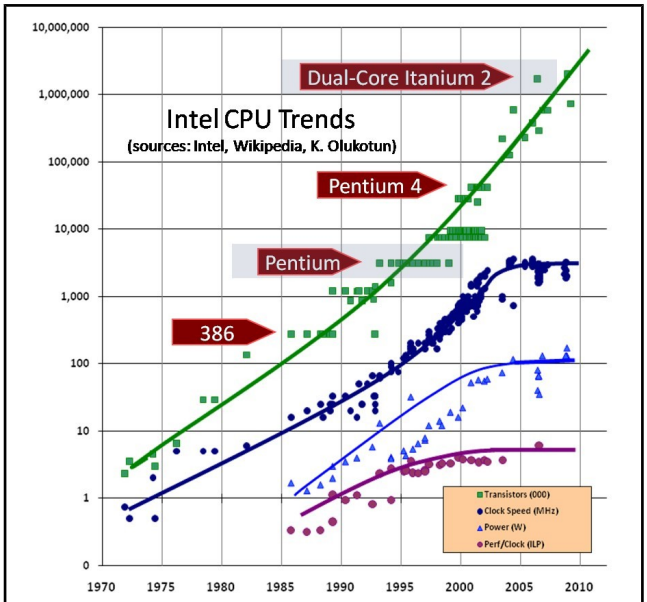

Figure 1: Moore's Law (credits: Intel.com)

The increased hardware capabilities enabled solving even harder problems. Multiple computation units enabled running multiple concurrent threads in the system. Intel's HyperThreading technology[6] allowed for faster and easier thread switching without the overhead of context switching.

3.1 Super Computers

Super computers take the hardware parallelism to the extreme. They constitute the fastest and most powerful computers in the world. They are very expensive and are used for computationally intensive applications. They do the computations very fast usually in sub-nano seconds

3.2 IBM Deep Blue

Deep blue is the first super computer to beat the world chess champion Garry Kasparov in a chess tournament. IBM researchers programmed the Deep blue to solve the complex, strategic game of chess. This made computer scientists to explore more on the capabilities of super computers. This gave way to scientists to come up with super computers to solve computationally intensive problems in many fields. Super computers are now widely used in analyzing large scale databases, molecular dynamics, financial modeling and risk analysis.

3.3 IBM Blue Gene

BlueGene/L was most the powerful super computer released by IBM in 2004. BlueGene was intended for biologists who can use to view the process of protein folding and gene development. This is a massive parallel computer with 65,536 nodes with system-on-a-chip technology (all components on a chip except RAM) with processing power of 360 teraFLOPS(trillion floating point operations per second). It has standard compilers and message passing environment and large memory space. Blue Gene/L followed energy-efficient design and computing model.

3.3.1 Architecture

Super computers generally have a very highly modular architecture. The smallest level nodes have 2 standard PowerPC 440 processing cores with a local memory of 2 GB. Each core has a PowerPC 440 FP2 core which is an enhanced "Double" 64 bit Floating-point unit. This supports the computational instruction and load and store instructions to be issued in parallel. The two processors in each node has two working modes co-processor mode and Virtual-node mode. In co-processor mode one processor handles computation and the other processor handles communication and in virtual-node mode both processors run user code, the computation and the communication load is shared between both the processors.

A shared non-cached memory is used to communicate between the tasks in the same node. There are some nodes dedicated for computation and some for IO. Each compute node has light weight OS to avoid system overhead. They use Linux OS.The kernel of the OS can run dual-threaded application process. The IO node use a Linux-based system and acts as an interface to the file system and other processes which are burden to compute nodes. This way minimum set of instructions are distributed among different processors. Since the nodes use Linux ,parallelism was simulated and tested in Linux clusters before implementing BlueGene/L.

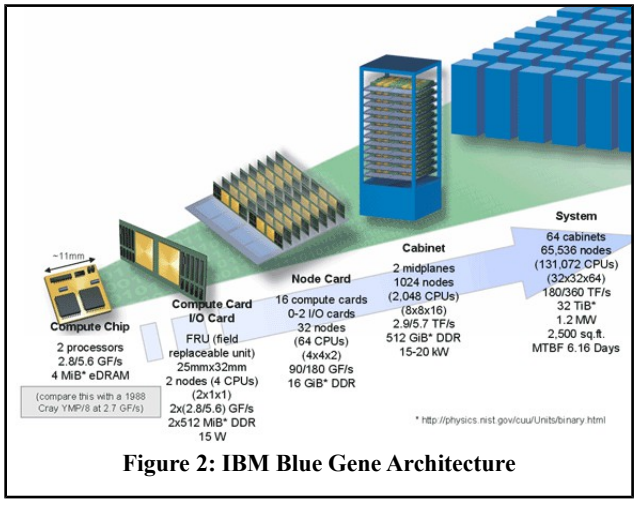

Figure 2: IBM Blue Gene Architecture

In blue gene/L the processor are interconnected through 3-D dimensional torus network. Each node in the network is connected to its six nearest neighbors. This makes interconnection among the nodes cost-effective as we don't need long cables to connect the nodes in each rack. Message passing is achieved with MPI message passing library. BG/L parallelizes simple tasks among many nodes and the nodes are dual-processor system. Each processor has extra floating point unit which increases the computational power of floating point operations. This makes them suitable for life sciences applications.

3.4 IBM Watson

IBM Watson is an AI intelligent system, that does Question-Answering posed in natural language. This was developed by the IBM DeepQA, an extension of the IBM super computer projects. In 2011, Watson famously beat the Jeopardy! Champion Brad Rutter and Ken Jennings received the first place price of 1 million dollars.

Watson was trained with 200 million pages from the web. Consuming multiple terabytes of information, including the full text of wikipedia. Watson is still considered one of the major milestones of Artificial Intelligence. Watson would not have been possible without the advent of IBM supercomputers.

3.5 Super Computer Applications

Apart from the flagship application of IBM Watson, Supercomputers have a wide variety of applications. IBM's Blue-gene was originally intended for computationally intensive biological applications. Especially, for the invisible thermodynamics and kinetics of the protein folding process. It was also used to find cure for diseases which creates mis-folded protein structure.

But today, they are used for [8]

- Recreating the Big Bang
  - The Super computer "mira" is being used to simulate the universe in the computer and understand how universe was created. This research can shed light on the origin of the universe.
- Understanding earthquakes
  - Earthquake modeling can predict when an earthquake occurs and save thousands of lives.
- Mapping Blood Stream
- Modeling Ebola virus/ Swine flu
  - Scientists are working on to find how different medicines prevents the ebola virus attaching to the human cell. Instead of working in a traditional lab and doing experiments in test tubes , the scientists are using software to make predictions. Super computers can evaluate millions of molecules in a day which is way better than the test tube experiments.
- Testing nuclear weapons
- Forecasting hurricanes.

4. PARALLELISM IN SOFTWARE

Parallelism in software is based on the data dependencies and control flow of the program. There are two types of parallelism which can be achieved in software level. One is data parallelism – different instructions operate on the same data and the other is Task level parallelism – multiple processes can be executed at the same time

4.1 MapReduce

Traditionally, highly scalable computation was available to only big companies, who invested heavily in building the massively parallel computers like the supercomputer discussed above. Recent evolution of software programming models have changed the way computations are parallelized and made the computation resources widely available. With the advent of MapReduce and other software breakthroughs, computation resources can be time-shared and hence the cost of the intense computation has dropped significantly.

4.2 MapReduce Architecture

MapReduce[1] is one of the game changing computation framework that has revolutionized the Information age. Organizations have realized the need for making sense of the vast amount of data available. This enormous amount of data that is being created is growing exponentially and the corporations are investing heavily to mine useful information out of the data and also to store and handle the data. To mine useful trends and patterns from the large amount of data efficiently we need to have a model which can handle and process large amounts of data in a reasonable amount of time. MapReduce is a programming model used to process and generate large datasets efficiently. This is invented by Google Inc. to handle inverted index computation in the Google search engine. Nowadays map reduce programming is widely used in many fields including machine learning, text processing.

Users choose the map and reduce function in the model and the model parallelizes the computation across large number of clusters and generates the output. The model splits the data and distributes it across the machines. It also chooses the scheduling algorithm for the program in the set of machines. Inter process communication across the machines and the failure handling is taken care by the model. Parallelization and distribution of the data is hidden from the user and so even users with no prior experience in parallelization and distributed systems can use it.

MapReduce can process terabytes of data on hundreds of thousands of machines. Map reduce roots from functional programming Haskell's lazy evaluation.

```
Hugs> map (*2) [1,2,3,4,5]
[2,4,6,8,10]
Hugs> foldl (+) 1 [1,2,3,4,5]
16
```

Table 1: Map-reduce as Haskell map & fold

In the specific example, the map function applies the multiplication operation on the list of all numbers which yields a new list [2,4,6,8,10].

The reduce function (fold) traverses through the list of numbers. It result. fold (+) 1 [1,2,3,4,5] which would result in 1+1 + 2 + 3 + 4 + 5, which is 16.

Though MapReduce is similar to the map-fold functions in Haskell, the important difference is that the Map and Reduce functions are applied across a large number of machines.

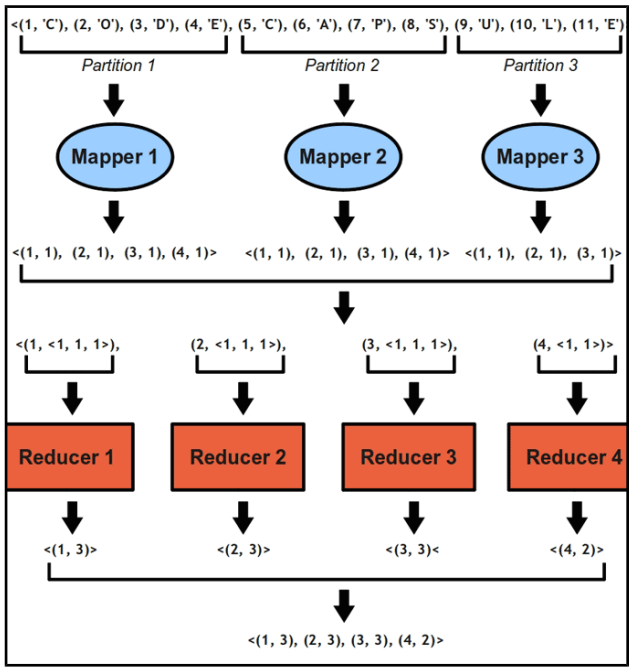

Figure 3: MapReduce : Word Count example (credits: google images)

The input data is structured as Key and Value [K, V] pairs. The map function gets the [K, V] as input and produces zero or more intermediate [K, V] pairs. The Map reduce follows key-based partitioning for all intermediate values with the same key are reduced together. Users can also specify the partitioning method and number of partitions. The reduce functions emits [K, V] as output.

Map-reduce is independent of the underlying file systems or the storage mechanism. As long as the problem can be split up into a map and reduce function, it can be computed using the map-reduce framework.

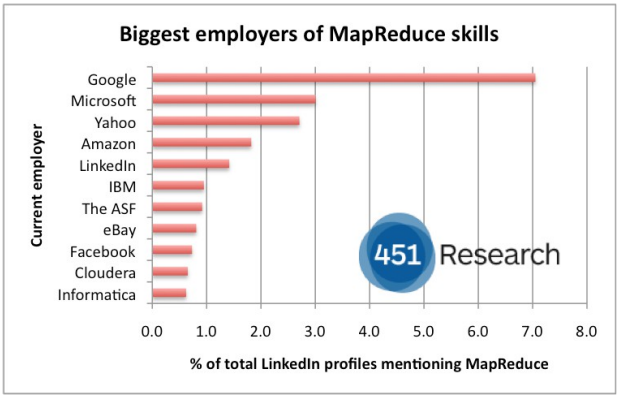

Figure 4: Employers seeking Mapreduce skills

5. APPLICATIONS

Parallel computing has seen a wide variety of applications in the last few decades. In this section, a few of the top applications are discussed in detail.

5.1 Search Engines

Search engines are an integral part of learning in this information age. The top search engines, Google and Bing index every page in the internet almost every passing minute. Google is estimated to have an index of over 100 Billion webpages.

Creating a big inverted index and serving 100B pages at query time with a latency of less than 100 ms is only possible with massively parallel computations. Both MicroSoft and Google have the world's top data-centers crawling the web and serving the search queries.

5.1.1 Indexing the web

As seen in the fig 5, the Indexing system constitutes one of the fundamental blocks of a Search engine. A MapReduce system is used to build an inverted index of the mirrored web documents.

The Map function takes input, the raw web documents and outputs a key-value pair. The key is every token in the document. Value is the document ID. The Reduce function takes input the token and list of doc ids and generates an output key value of the token to a sorted list of all doc ids. One important thing to note is that this system can scale very well with the number of webpages. Even if the number of webpages increases 100 fold in the future, with additional computational units, this indexing scheme can compute the index in a reasonable amount of time.

5.1.2 PageRank – scoring the web

PageRank is Google's secret sauce to rank the webpages in the web. Each of the 100Billion web pages are ranked with reference to each other in a massively parallel graph-computation.

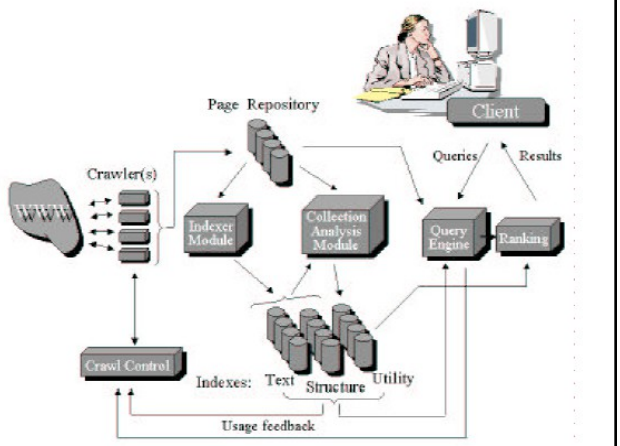

Figure 5: Architecture of Search Engine (Credits: Google Images)

Each of the webpages are treated as nodes(V) in a graph and a hyperlink is considered an edge(E). This link graph has over 100Billion nodes and has far fewer edges. The PageRank algorithm assigns a numerical score to each page in a way of measuring its relative authority within the set.

PageRank is an iterative algorithm based on the link-graph. Let us assume cnn.com and bbc.co.uk are authoritative sources. Each of the out-links of these pages (A, B, C), receive some of the "weights" from the authoritative sources. In the next iteration, the out-links of these pages in-turn get weights from A, B, C. In essence, each page receives a PageRank based on the incoming links. It can be proved that this iterative algorithm converges to a stationary value. PageRank algorithm can be formulated as a MapReduce to compute effectively.

5.2 Google Brain – Deep learning

Google Brain[12] is a deep learning research project at google. This aims to build large-scale neural networks with multiple level of hidden layers, which imitates human brain. Each layer has number of interconnected nodes. The input from input layer passes through many hidden layers. The computation is done in the hidden layers by iterating on the weights of different layers and the hidden layer gives output to the output layer. We can train and build a model on how an object looks and how it sounds like.

To train such a model to predict an object we need large scale distributed deep-networks. The multiple hidden layers will allow us to compute very complex features of the input. Billions of parameters are trained using tens of thousands of CPU cores. A software framework called DistBelief[12] achieves model parallelism within a machine using multi-threading and between machines using message passing. This framework also offers data parallelism. There are 2 methods of large scale distributed training used (1) Downpour SGD (2) Sandblaster L- BFGS.

5.3 Graph Partitioning

A Graph is one of the most used data-structures to represent data and their dependencies. Graph G has vertices V={v1,v2...vn} and edges E ={e1,e2,..en}. Two vertices are connected with an edge if there exists dependencies.
Data which can be represented in the form of Graph G with vertices V and edges E can be partitioned to smaller components of graph. This is called graph partitioning. Usually the graph is split in a way the number of edges connecting the split components is small. Graph partitioning[2] is used to divide the data and tasks so that tasks and data can be run in parallel. Parallelization of workload depends on how independent they are from each other. Graph partitioning the data in a way there are less dependencies will hep to achieve more parallelization . It is widely used in scientific computing, scheduling of tasks in multi-processor systems.

5.3.1 Partitioning Models

Once we have the data represented as graph. Our goal is to divide the vertices V of graph G into k partitions where k is the number of processors. Also we need to make sure the number of edges between separated partitions is less, edges represent intercommunication between partitions. The partitions are then distributed to the processors and the processors are run in parallel.

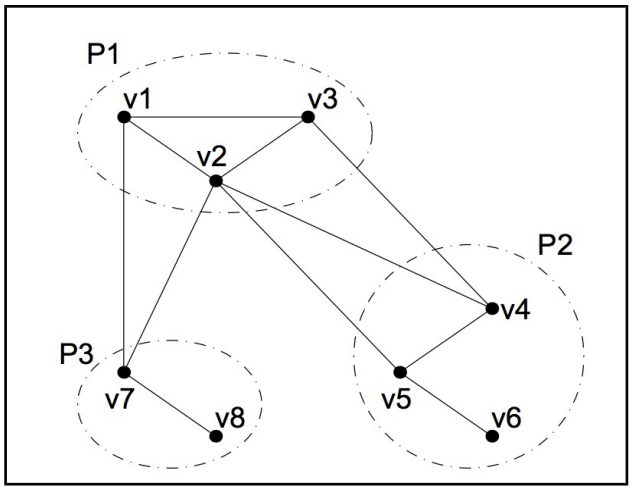

Figure 6: Graph Partitioning

There are different kinds of graph partitioning methods used to achieve parallelism.

5.3.2 Standard graph partitioning

The vertices of the graph is partitioned into equally weighted sets and the weight of the edge connecting the sets should be minimal**.** It follows edge cut method. It is based on the concept the number of edges cut in the graph is equal or proportional to the volume of the communication lost.

The objective of the standard partitioning has many flaws. The ovals in the figure represent the processors. Each vertex has data of one unit and edge connecting 2 vertices has weight of 2 data units communication one unit of data in each direction. In the given figure it is assumed the edges between the vertices (v3,v4), (v2,v4),(v2,v5),(v2,v7),(v1,v7) are cut and the total communication loss is 10 units as each edge communicates 2 data units. However the data communicated from vertex v7 in processor P3 to v2 in processor P1 is same as data from vertex v7 in processor P3 to v1 in processor P1. So the data can be communicated only once and this is the same with (v2,v4),(v3,v4) and (v4,v2),(v4,v3). So the total communication is lost in edge cut method is 7.

Standard partitioning method considers only the volume of the message communicated and it doesn't count the number of the message sent. The total volume of the message and the total number of messages both should be considered while partitioning the workload.

Usually the performance of a parallel application is hampered by the slowest processor. Even though we distribute the workload evenly among different processors, this doesn't guarantee the communication between the nodes is balanced. Standard partitioning focusses to minimize the total volume of messages rather that the number messages handled by one processor.

Usually the time taken to communicate between nodes depends on the distance between the sending and receiving node. Here the distance implies the number of switches the message has to cross to reach the destination. It is the best practice to limit the communication to adjacent or processors nearby. This will increase the throughput and avoid message contention.

With all the limitations of the edge cut metric, this partitioning method is proved to be successful for grid based problems and parallel solution of differential equations.

The graph partitioning model is an undirected graph model and it cannot express asymmetric dependencies. The model works well for only data with symmetric dependencies and symmetric partitions. It doesn't allow input and output partitions to be different.

5.3.3 Bipartite graph model

This model overcomes the limitations of standard graph partitioning. This approach can be used on data with asymmetric dependencies. In Bipartite graph G=(V1,V2,E) the vertices are split into two disjoint subsets V1 and V2 with $E \subset V1xV2$. Any edge in the dataset should not connect to more than one vertex in the other dataset. This model can be applied to problems or data where standard model is not a fit. This partition allows input and output vertices to be different and initial partition and final partition to be different. It addresses the edge cut metric flaw of standard partitioning model by allowing only one edge to connect from one dataset to other dataset.

6. CONCLUSION

This paper discussed the two ways in which parallel computation can be achieved – hardware and software. The hardware solution is more expensive but provides absolute control over the architecture of the parallel computation, while the software solution is more generic and flexible. The additional flexibility, low cost and the affordance of software solution has driven the industry to great heights in the last decade.

The paper also discussed a few applications in the area. Building a web-index and computing PageRank were discussed in detail. Building large scale deep neural networks like Google Brain has been made possible only with the advent of the massively scalable clusters at Google. In conclusion, parallel computing has changed the world and the author believes that this is just the beginning.

7. REFERENCES

[1] Dean, J., & Ghemawat, S. (2008). MapReduce: simplified data processing on large clusters. Communications of the ACM, 51(1), 107-113.

[2] Hendrickson, B., & Kolda, T. G. (2000). Graph partitioning models for parallel computing. Parallel computing, 26(12), 1519-1534.

[3] Adiga, N. R., Almási, G., Almasi, G. S., Aridor, Y., Barik, R., Beece, D., ... & Kurhekar, M. P. (2002, November). An overview of the BlueGene/L supercomputer. In Supercomputing, ACM/IEEE 2002 Conference (pp. 60-60). IEEE.

[4] Batcher, K. E. (1980). Design of a massively parallel processor. Computers, IEEE Transactions on, 100(9), 836-840.

[5] http://www-03.ibm.com/ibm/history/ibm100/us/en/icons/bluegene/

[6] Ge, Steven, Xinmin Tian, and Yen-Kuang Chen. "Efficient multithreading implementation of H. 264 encoder on Intel hyper-threading architectures."Information, Communications and Signal Processing, 2003 and Fourth Pacific Rim Conference on Multimedia. Proceedings of the 2003 Joint Conference of the Fourth International Conference on. Vol. 1. IEEE, 2003.

[7] http://www.cs.ucr.edu/~mart/CS260/IBM-BlueGene-Torus-Journal.pdf

[8] http://www.livescience.com/6392-9-super-cool-supercomputers.html

[9] Ferrucci, David A. "Introduction to “this is watson”." *IBM Journal of Research and Development* 56.3.4 (2012): 1-1.

[10] *IBM's "Watson" Computing System to Challenge All Time Greatest Jeopardy! Champions*, Sony Pictures Television, 2010-12-14, archived from the original on 2013-06-06, retrieved 2013-11-11

[11] Page, Lawrence, et al. "The PageRank citation ranking: Bringing order to the web." (1999).

[12] Dean, Jeffrey, et al. "Large scale distributed deep networks." *Advances in Neural Information Processing Systems*. 2012.