OUTLOOK ON VARIOUS SCHEDULING APPROACHES IN HADOOP

P. Amuthabala

Assistant professor,
Dept. of Information Science & Engg.,
Atria Institute of Technology
Bangalore, India
amuthabala@yahoo.com

Kavya.T.C

student,
Dept. of Information Science & Engg.
Atria Institute of Technology
Bangalore, India
kavyabharu@gmail.com

Kruthika.R

student,
Dept. of Information Science & Engg.
Atria Institute of Technology
Bangalore, India
kruthi944@gmail.com

Nagalakshmi.N

student,
Dept. of Information Science & Engg.
Atria Institute of Technology
Bangalore, India
nagalakshmibe04@gmail.com

ABSTRACT

MapReduce is used for processing and generating sets large data .A open source framework of MapReduce is Hadoop [1]. MapReduce and Hadoop represent a good alternative for efficient large scale data processing and advanced analytics in an enterprise. In Heterogeneous computing, map or schedule a processor to a single core or different type of processors to a single core or a processor to many cores or many processors to many cores. So the usage of heterogeneous multi-core processors for the efficient performance in map reduce environments is increasing. Therefore the single heterogeneous multi-core processors consists of small and big cores where small cores provide power efficient and big-cores provide high-performance , which includes Inductive Logic Programming and Multilayer Perception to be extracted dynamically. This paper addresses various scheduling approaches that helps in improving the performance in heterogeneous environment.

The outcome of this paper shows that traditional approaches used in Hadoop suffers from various issues. This paper will encourage in addressing those issues and discusses various scheduling approaches which helps in big data analysis.

I. INTRODUCTION

Big data is an evolving term that describes any large amount of structured, semi-structured and unstructured data that has the potential to be mined for information. Big data can be characterized by 3 Vs: the extreme volume of data, the wide variety of types of data and the velocity at which the data must be must processed. Because big data takes too much time and costs too much money to load into a traditional relational database for analysis, new approaches to storing and analyzing data have emerged that rely less on data schema and Data quality stead, raw data with extended meta data is aggregated in a data lake and machine learning and artificial intelligence programs use complex algorithms to look for repeatable patterns. Big Data Analytics is often associated with cloud computing because the analysis of large data sets in real time requires a platform like Hadoop to store large data sets across a distributed cluster and Map Reduce to coordinate, combine and process data from multiple sources.

MapReduce is used for processing and generating large data sets. It is basically a processing system, used to process batch data. But, Interactive analysis are not properly deployed in Map Reduce. Hadoop [1] is an open source framework of MapReduce. The evolution of Hadoop is beyond batch processing. It is a software that allows parallel computing. It is mainly used to storing and processing big data in a distributed environment. It is a databases which stores data and it is of NoSQL type. It can be used in a servers to thousands of machines, which enables each of this to offer local computation and storage.

Hadoop consists of Hadoop Distributed File System (HDFS) and Hadoop MapReduce.

1) Hadoop Distributed File System (HDFS) stores large amount of Data, it provides high throughput for accessing the data on the clusters and 2) Hadoop MapReduce is a software framework for processing of data.

Heterogeneous computing is typically used to map or schedule a processor to a single core or different type of processors to a single core or a processors to many cores or many processors to many cores. So we use heterogeneous multi-core processors for the efficient performance in map reduce environments. This paper describes about performance Impact Estimation which is a mechanism which provides workload-to-core mapping would give the best performance. So the PIE would require CPI stacks, ILP and MLP to predict about the performance, if it requires to run on different scheduling's. PIE can upgrade system performance by average of 5.5% over the state-of-the-art scheduling and by 8.7% over a Sampling based scheduling policy. This paper also show that PIE would useless hardware's.

There are many algorithms and techniques applied in improving the performance of data processing and based changing needs and growing data many techniques are being evolving and this help in fulfilling the growing needs of data.

Section 1 of this paper discuss about the background of map reduce and Hadoop. Section

2 presents about the various techniques used in Hadoop and there drawbacks which payed way to new techniques. And finally section 3 provides the conclusion and acknowledgement.

II. MAP REDUCE

Map reduce is used in data intensive application. It improves the performance by providing automatic data management, fault detection and recovery. It helps in processing large amount of data in economical beneficial way. It is an important processing model. Map reduce can take advantage of locality of data, processing it on or near the storage assets in order to reduce the distance over which it must be transmitted.

MapReduce is a programming model [2] and an associated implementation for processing and generating large data sets. Users specify map function that processes a key/value pair to generate a set of intermediate key/value pairs, and a reduce function that merges all intermediate values associated with the same intermediate key.

Map Reduce pipeline [3] the main computation is expressed as two user-defined functions: map and reduce. The map function takes an input pair and produces a list of intermediate key/value pairs. The reduce function then merges or aggregates all the values associated with the same key. MapReduce jobs are executed across multiple machines: the map stage is partitioned into map tasks and the reduce stage is partitioned into reduce tasks.

The execution of each map (reduce) task is comprised of a specific, well-defined sequence of processing phases (see Fig. 1).Map task consists of the following generic phases: 1. Read phase – a map task typically reads a block (e.g., 64 MB) from the Hadoop distributed file system (HDFS). 2. Collect phase – this generic phase follows the execution of the map phase with a user-defined map function. 3. Spill phase – we measure the time taken to locally sort the intermediate data and partition them for the different reduce tasks, applying the combiner if available, and then writing the intermediate data to local disk. 4. Merge phase – we measure the time for merging different spill files into a single spill file for each destined reduce task. Reduce task consists of the following generic phases: 1. Shuffle phase – we measure the time taken to transfer intermediate data from map tasks to the reduce tasks and merge-sort them together. We combine the shuffle and sort phases because in the Hadoop implementation, these two sub phases are interleaved. The processing time of this phase depends on the amount of intermediate data destined for each reduce task and the Hadoop configuration parameters.

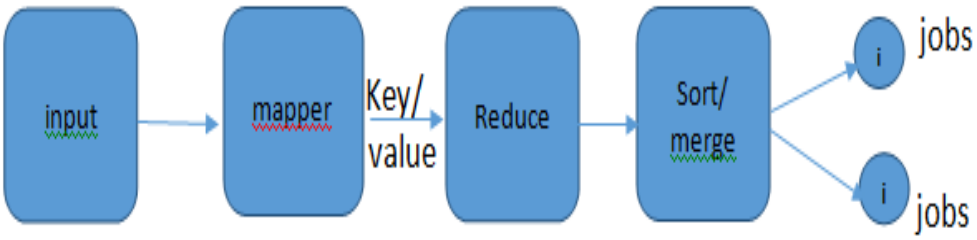

Fig 1. Map Reduce Process

III. Hadoop

It is the open source framework on Map Reduce. It is a Java based programming framework. It helps in processing of large set of data in a distributed environment. Hadoop-Map Reduce is a powerful technique which is based on parallel processing for big data analysis on distributed environment of hardware clusters. The evolution of Hadoop is beyond batch processing. It is a Software that allows parallel computing. It is mainly used to storing and processing big data in a distributed environment. It is a databases which stores data and it is of NoSQL type. It can be used in a servers to thousands of machines, which enables each of this to offer local computation and storage.

Hadoop consists of Hadoop Distributed File System (HDFS) and Hadoop MapReduce.

1) Hadoop Distributed File System (HDFS) stores large amount of data, it provides high throughput for accessing the data on the clusters and

2) Hadoop MapReduce is a software framework for processing of data.

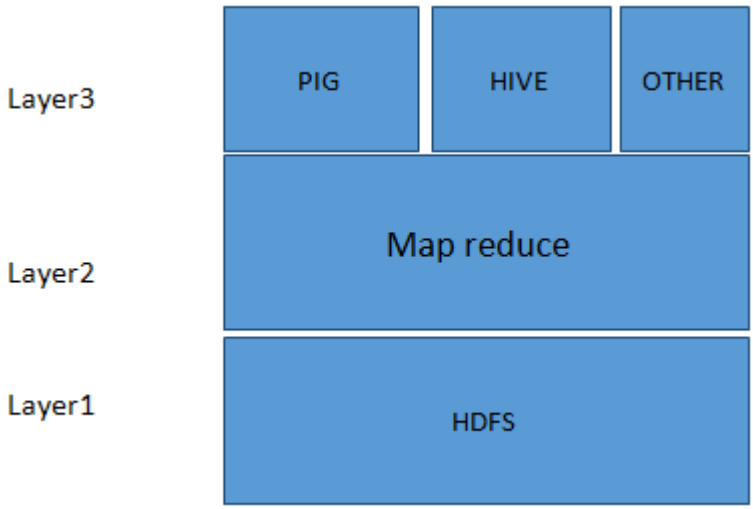

Fig.2 Hadoop architecture

IV. SCHEDULERS IN HADOOP

Scheduler will be used to carry out many activities of scheduling. We use scheduling in heterogeneous multi core processors which gives high and efficient performance by having big or small cores. Some of those include:

1) Sampling Based Scheduling

2) Performance Impact Estimation Scheduling (PIE)

a) Random Scheduling

b) Memory Dominance Scheduling

3) Dynamic Scheduling

4) Dynamic PIE scheduling

Other schedulers:

1) Bias Scheduling

2) HASS Scheduling

3) Age-based Scheduling

Sampling Based Scheduling:

This can be used to examine which scheduling policy can be directly approached for heterogeneous multi-cores. It maps different workload-to-core at runtime and selects the best. The main disadvantage is increasing overhead due to sequential moving of workloads among different types of core.

Performance Impact Estimation (PIE) Scheduling:

It calculates or estimates the performance of the workload on different core type.

Components:

1. CPI stacks
2. Base component
3. Memory component
4. ILP (Inductive Logic Programming)
5. MLP (Multilayer Perception)
6. Big and Small cores

We predict MLP in the memory CPI component using bias cores and small cores. Workload-to-core mapping using PIE Scheduling compares itself with Random Schedulers. Scheduling and Memory-dominance scheduling, where Memory-dominance scheduling involves scheduling memory instance workloads to the small core. Sometimes without considering the ILP, PIE uses only the MLP for efficient performance as the MLP-ratio scheduling. Advantage of PIE Scheduling is found across both cores in which workload-to-core mapping scheduler gives the highest system throughput. As it is solved in static setting which means that the entire execution will be done on the single core specified which ignore efficiency.

Dynamic Scheduling:

It improves PIE by dynamically adjusting to the behaviour of workload phase. This overcomes the static state of PIE scheduling. Overhead can be avoided by- shared LLC, private powered-off LLC, private powered-on LLC.

Dynamic PIE Scheduling:

PIE Scheduling can be applied to any core type which has any number of cores. To have optimal efficiency it collects: CPI stacks, number of misses, number of dynamically executed instructions, inter-instruction dependency on big core, hardware and software. This is scalable as it is dynamic but Profile information cannot be collected on hardware. Collecting CPI stacks is sometimes complicated which results in missing of event.

Apart from above,

Bias Scheduling:

Same as memory-dominance scheduling. It performs and schedules programs which has memory on small core and domination of execution cycles on big core.

HASS Scheduling:

Used for scalability of programs on cores, which is static. This does not consider time varying behaviour of execution.

Age based scheduling:

The remaining mapping or execution can be done in this scheduling.

V. APPROCHES IN IMPROVISING SCHEDULERS

A. PIG

Map reduce leads to hacking multi input flows and implementing the operations repeatedly. This slows down the data analysis and processing difficult. To address the above problem Pig systems are developed here which contains high level data manipulation methodology like SQL by retaining standard functions of map reduce systems.

Pig [4] compares the programs written in programming language called pig Latin. This Latin program used as the input to the pig on Hadoop map reduce jobs.

Heterogeneous multicore architectures [6] provide more energy/area efficiency compared to homogenous. The efficiency is determined if the OS assigns applications to appropriate cores. Such heterogeneity algorithms was proposed earlier but didn't work on larger cores as left some threads which were unsolved. So, here we introduce a scheme to overcome this problem by providing the information which is needed to make a proper assignment is provided to job itself. Scheduling on heterogeneous multicore system based on architectural signatures [6] is used to solve the above issue.

This architectural signatures are just the summery of application's architectural properties like memory-boundedness, available ILP etc. This signature is generated offline and is bounded to application binary so that OS determines the best thread to core assignment for increasing performance. Clusters had to be shared by different users, the problem arises in scheduling and data locality. The fair scheduling and data locality is achieved through delay scheduling algorithm [7]. Through this algorithm we can achieve optimal data locality and through put is increased by 2x which rather increases fairness in scheduling

B. LATE

LATE [8]: straggles are identified as early as possible and response time is reduced using LATE algorithm. LATE prioritizes tasks to speculate, selecting fast nodes to run a capping speculative tasks to prevent from thrashing. Late improves the response time of map reduce jobs by a factor of 2 in large clusters of EC2.LATE is called for longest approximate time to end 1st and executes tasks which improves job response time.

The performance of this LATE is very poor because of static manner which affects the progress of the compute task, hence this is inefficient in heterogeneous environment.

C. SAMR

SAMR [5] (A Self adaptive Map Reduce scheduling algorithm.): SAMR is an algorithm which plays a vital role in heterogeneous environments. It capably decreases the execution time required for the Map Reduce jobs. The growth of each task will be calculated by SAMR, this helps it to adapt to continuously varying environment straightforwardly. When a job is submitted to this algorithm, it splits the job into Map task and Reduce task. This task will be then assigned to nodes. The historical information associated with the nodes are read by this Scheduling Algorithm, this helps in estimating the time required for the map and the reduce task and also to classify the nodes that are slow into map slow nodes and reduce slow nodes. This classification helps in useful utilization of the resource.

D. ARIA

ARIA [9] (Automatic Resource Inference and Allocation): This helps automatically tailors and controls resource allocations to different applications for achieving their performance goals. ARIA consists of 5 components Job profile, Profile Database, Slot Allocator, SLO scheduler. This uses earliest deadline first algorithm and the resource requirements are enforced by SLO scheduler. ARIA consists of 5 components Job profile, Profile Database, Slot Allocator, SLO scheduler.

1. Job Profiler: The information about the job will be collected and a job profile is prepared for the jobs that are running at present or for the jobs that have been completed.

2. Profile Database: My SQL database is used to store history of the jobs.

3. Slot Estimator: It calculates the smallest number of map and reduce slots desirable job based on the time limit and the profile of the job. This helps in achieving the Service Level Objective.

4. Slot Allocator: It allocates the task to various jobs using the information obtained by the slot estimator.

5. SLO-Scheduler: It is a vital component which harmonizes actions between all the other components of the ARIA. This uses earliest deadline first algorithm and this also enforces the resource needs for the jobs.

When a job is submitted, based the time limit of the job the ARIA will allocate the

Suitable number of map and reduce slots.

E. SimMR

SimMR [10]: it is a simulation environment. It helps in resource allotment for the jobs that are running currently and the jobs that have been submitted. It also plays a vital role in estimating about the various scheduling approaches. SimMR consists of three components

1. Trace Generator: this will create the workloads for the map reduce task that are replay able.

2. Simulator Engine: It embeds the functions of the job master in Hadoop and

3. Pluggable scheduling policy: It helps in job ordering and allocation of resource for various jobs.

SimMR can replicate complex workloads in seconds and process millions of events in a second.

F. Adaptive Scheduler

Adaptive Schedulers [11]: This scheduler will efficiently allocate the resources to various jobs dynamically. This job allocation process will be based on the completion time of the job. The design of this scheduler helps in achieving the performance goals of the user at a higher level. It is can schedule the resources for multiple jobs. The Map reduce cluster is classified into two pools by the scheduler, considering the properties of the jobs and there time limit the jobs will be allocated to the pools. The performance of the job depends on the pool to which it is allocated. Based on the specific hardware code and the performance of the jobs, the code to be run on the hardware available will be decided by the scheduler. This scheduler provides dynamic resource allotment and whenever possible provides hardware affinity. This helps in achieving the high level performance for the application even in the occurrence of the hybrid systems.

G. TARAZU

TARAZU [12]: It helps in improving the Map Reduce performance in heterogeneous environment. TARAZU has 3 components:

1. Communication aware load balancing of map computation (CALB).

2. Communication aware scheduling (CAS)

3. Predictive load balancing of reduce computation (PLB)

CALB begins the map task. This will be depend on the map/shuffle process is important.

CAS: it will extend the remote task. This will be performed throughout the map phase.

PLB: It will lessen the load balance across the heterogeneous nodes.

It addresses the two big issues of Map Reduce 1. The load balancing process used in Map Reduce results in network problems and 2. There is a load imbalance due to heterogeneity.

H. SAILFISH

SAILFISH [13] is a Map Reduce frame work. It is for used for large data processing.

This helps in handling the intermediate data generated by the map task, which are later used by the s helps reduce task. This helps in enhancing the performance of the Map Reduce task. Sailfish design is firmed on the intermediate nodes. It is focused for the data generated by the map task and that are later used by the reduce task. Sailfish uses Intermediate data files (I file) to obtain the data from the map task, and then given to the reduce task. The particular number of reduce task will be determined by the Sailfish, this helps in parallel and automatic execution. This framework plays a vital role in improving the performance.

I. PIKACHU

TARAZU [11] estimates load balance before map reduce job execution which will not optimise the mad reduce by rebalancing so PIKACHU [14] was introduced. PIKACHU is a task scheduler which optimises the map reduce on Hadoop and TARAZU. This uses dynamic load balancer which helps in partitioning keys for fast and slow nodes at run time and tells the reducer to perform the task.

J. TROUGHPUT SCHEDULER

TROUGHPUT SCHEDULAR [15], it schedules jobs uniquely by leaving both server capabilities and job task parameters. This optimizes the task assignment thereby reducing the overall execution time.

CONCLUSION

Single-ISA heterogeneous multi-core processors has as small and big cores for their respective performances. As the performance is the main aspects of any problem or solution, single-ISA heterogeneous multi-core processors which uses different core types on a single processor can be used. It also has the potential to improve energy and efficiency without giving up the significant performance

Above mentioned schedulers improves scalability for itself, improves multithreaded workload. Uses different core types and processors

By using all this concepts it gave a full coordination to implement different schedulers which gave its best for the mapping of schedulers to processor's workload. The scheduling can also be expanded for optimization of fairness. This would be the ongoing work considered in future.

Map reduce programming is not a good match for all the problems, Data security, Data management and governance. In spite of all this challenges Hadoop promise a low cost high availability and processing power.

In the future, once the servers with heterogeneous multi-core processors become available, we plan to conduct more testbed experiments and a variety of job ordering scheduling policies for achieving fairness guarantees or job completion objectives.

REFERENCES:

[1] T. White, Hadoop:The Definitive Guide. Yahoo Press.
[2] J. Dean and S. Ghemawat, "MapReduce: Simplified data processing on large clusters," Communications of the ACM, vol. 51, no. 1, 2008.
[3] Z. Zhang, L. Cherkasova, and B. T. Loo, "Benchmarking approach for designing a mapreduce performance model," in ICPE, 2013, pp. 253–258.
[4] A. Gates, O. Natkovich, S. Chopra, P. Kamath, S. Narayanam, C. Olston, B. Reed, S. Srinivasan, and U. Srivastava, "Building a highlevel dataflow system on top of mapreduce: The pig experience," PVLDB, vol. 2, no. 2, pp. 1414–1425, 2009.
[5] Q. Chen, D. Zhang, M. Guo, Q. Deng, and S. Guo, "Samr: A self-adaptive mapreduce scheduling algorithm in heterogeneous environment," in IEEE 10th International Conference on Computer and Information Technology (CIT), 2010.
[6] D. Shelepov and A. Fedorova, "Scheduling on heterogeneous multicore processors using architectural signatures," in Proceedings of the Workshop on the Interaction between Operating Systems and Computer Architecture, 2008.
[7] M. Zaharia et al., "Delay scheduling: A simple technique for achieving locality and fairness in cluster scheduling," in Proceedings of EuroSys, 2010.
[8] M. Zaharia et al., "Improving mapreduce performance in heterogeneous environments," in Proceedings of OSDI, 2008.
[9] A. Verma, L. Cherkasova, and R. H. Campbell, "ARIA: Automatic Resource Inference and Allocation for MapReduce Environments," in Proc. of ICAC, 2011.
[10] ——, "Play It Again, SimMR!" in Proceedings of Intl. IEEE Cluster'2011.
[11] J. Polo etal., "Performance management of accelerated mapreduce workloads in heterogeneous clusters," in Proceedings of the 41st Intl. Conf. on Parallel Processing, 2010.

[12] F. Ahmad et al., "Tarazu: Optimizing MapReduce on Heterogeneous Clusters," in Proceedings of ASPLOS, 2012.
[13] S. Rao et al., "Sailfish: A Framework for Large Scale Data Processing," in Proceedings of SOCC, 2012.
[14] R. Gandhi, D. Xie, and Y. C. Hu, "Pikachu: How to rebalance load in optimizing mapreduce on heterogeneous clusters," in Proceedings of 2013 USENIX Annual Technical Conference. USENIX Association, 2013.
[15] G. Gupta, C. Fritz, B. Price, R. Hoover, J. DeKleer, and C. Witteveen, "ThroughputScheduler: Learning to Schedule on Heterogeneous Hadoop Clusters," in Proc. of ICAC, 2013.

AUTHORS PROFILE

P. Amuthabala has obtained her B.E in Computer science Engineering at Avanishilingam University in Coimbatore, Tamilnadu, India in 2002 and her M.E degree in Software Engineering at Bangalore University in Bangalore, Karnataka, India in 2011.She is working as an Assistant Professor in Information Science Department at Atria Institute of Technology, Bangalore, Karnataka. Her research areas of Interest include Data Mining Data warehousing and Cloud Computing. She is currently doing her PhD in Karpagam University.

Kruthika.R Doing B.E in Information Science and Engineering at Atria Institute of Technology, Visvesvaraya Technology University, Bangalore.

Kavya.T.C Doing B.E in Information Science and Engineering at Atria Institute of Technology, Visvesvaraya Technology University, Bangalore.

Nagalakshmi.N Doing B.E in Information Science and Engineering at Atria Institute of Technology, Visvesvaraya Technology University, Bangalore.