ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 6, Special Issue 2, February 2017

National Conference on Recent Advancements in Engineering & Technology and Management 2017 (NCRAETM 2K17)

18th February 2017

Organized by

Sree Rama Engineering College (SRET), Tirupati- 517 507, India

Scheduled Traffic Alert Division and Accumulation in Various Multiple Hub Processors

Ramisetti Lokesh[1], N. Jaya Krishna[2]

M.Tech, Department of Computer Science and Engineering, Sreerama Engineering College, Tirupati, AP, India [1]

Assistant Professor, Department of Computer Science and Engineering, Sree Rama Engineering College Tirupati, AP, India [2]

Abstract: We be trained to cut back network site visitors fee for a MapReduce job via designing a novel intermediate knowledge partition scheme. Additionally, we jointly consider the aggregator placement obstacle, the place each aggregator can scale down merged traffic from more than one map duties. A decomposition-situated dispensed algorithm is proposed to take care of the tremendous-scale optimization challenge for tremendous information software and an internet algorithm is also designed to adjust information partition and aggregation in a dynamic method. In the end, vast simulation outcome demonstrate that our proposals can enormously scale back community traffic rate below both offline and online circumstances. In this, we prototype and review a new Hadoop scheduler, known as DyScale, that exploits capabilities provided by heterogeneous cores within a single multi-core processor for reaching a type of performance goals. A average MapReduce workload comprises jobs with unique performance ambitions: giant, batch jobs which might be throughput oriented, and smaller interactive jobs that are response time sensitive. Heterogeneous multi-core processors permit growing virtual useful resource swimming pools based on gradual and quick cores for multi-class priority scheduling. Due to the fact that the equal data may also be accessed with either sluggish or speedy slots, spare assets (slots) can be shared between one of a kind useful resource pools. Using measurements on an exact experimental setting and through simulation, we argue in prefer of heterogeneous multi-core processors as they acquire faster(as much as 40%) processing of small, interactive MapReduce jobs, even as offering multiplied throughput (up to 40%) for colossal, batch jobs. We overview the performance advantages of DyScale versus the FIFO and capability job schedulers which are commonly used within the Hadoop community.

KEYWORDS: MapReduce; Hadoop; heterogeneous systems; scheduling; performance; power.

1. INTRODUCTION

MapReduce and its open supply implementation Hadoop offer a scalable and fault-tolerant framework for processing large information units. MapReduce jobs are robotically parallelized, dispensed, and performed on a large cluster of commodity machines. Hadoop used to be at the beginning designed for batch-oriented processing of massive construction jobs. These purposes belong to a category of so-referred to as scale-out applications, i.E., their completion time may also be elevated with the aid of utilising a greater quantity of assets. For illustration, Hadoop customers practice a simple rule of thumb: processing a gigantic MapReduce job on a double dimension Hadoop cluster can lessen job completion in half of. This rule is relevant to jobs that have to process colossal datasets and that include a colossal quantity of tasks. Processing these tasks on a higher number of nodes (slots) reduces job completion time. Efficient processing of such jobs is throughput-oriented and may also be tremendously increased with additional scale-out assets. When more than one customers share the same Hadoop cluster, there are a lot of interactive advert-hoc queries and small MapReduce jobs which might be completion-time touchy. Additionally, a developing number of MapReduce applications (e.G., customized promoting, sentiment evaluation, spam detection) are closing date-driven, as a consequence they require completion time guarantees. To beef up the execution time of small MapReduce jobs, one can't use the scale-out approach, however could benefit making use of a scale-up approach, where duties execute on faster assets. There is a list of interesting opportunities for making improvements to MapReduce processing

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 6, Special Issue 2, February 2017

National Conference on Recent Advancements in Engineering & Technology and Management 2017 (NCRAETM 2K17)

18th February 2017

Organized by

Sree Rama Engineering College (SRET), Tirupati- 517 507, India

furnished by using heterogeneous processor design. To begin with, both speedy and gradual Hadoop slots have the identical entry to the underlying HDFS data.
This eliminates the information locality disorders that might make heterogeneous Hadoop clusters created from quick and sluggish servers1 being inefficient . Nonetheless, when every node contains heterogeneous core processors, then any dataset (or any job) can be processed via both fast or gradual digital resource swimming pools, or their mixture. second, the likelihood of task (job) migration between sluggish and speedy cores enables bettering efficiency guar- static frameworks without the process migration characteristic. Among the many challenges are i) the implementation of recent mechanisms in help of dynamic resources allocation, like migration and digital useful resource pools, ii) the help of accurate job profiling, notably, when a job/venture is accomplished on a mix of speedy and sluggish slots, iii) the analysis of per job performance exchange-offs for making the correct optimization choices, and iv) increased administration complexity.

II. MAPREDUCE BACKGROUND

Within the MapReduce model, computation is expressed as two features: map and minimize. MapReduce jobs are completed across a couple of machines: the map stage is partitioned into map tasks and the decrease stage is partitioned into shrink duties. The map and cut back duties are done through map slots and shrink slots. In the map stage, every map assignment reads a break up of the input knowledge, applies the user-outlined map function, and generates the intermediate set of key/price pairs. The map task then types and partitions these data for one-of-a-kind scale down tasks in keeping with a partition operate. Within the cut back stage, each reduce undertaking fetches its partition of intermediate key/value pairs from all of the map tasks and kinds/merges the info with the identical key. After that, it applies the consumer-defined cut back operate to the merged worth list to produce the aggregate results (that is known as the slash phase). Then the diminish output is written back to a disbursed file method. The assignment of tasks to slots is done in a grasping approach: assign a challenge from the chosen job instantly whenever a worker reports to have a free slot. Even as, a knowledge locality consideration is taken under consideration: if there is a choice of available slots within the process to be allotted to job, then the slots which have information chunks of job locally available for processing are given precedence . If the number of duties belonging to a MapReduce job is larger than the total number of processing slots, then the challenge undertaking takes a couple of rounds, that are referred to as waves.

III. DYSCALE FRAMEWORK

We advocate a brand new Hadoop scheduling framework, referred to as DyScale, for effective job scheduling on the heterogeneous multi-core processors. First, we describe the DyScale scheduler that permits developing statically configured, committed virtual useful resource pools situated on special types of available cores. Then, we reward the improved version of DyScale that allows the shared use of spare assets amongst existing virtual useful resource pools.

A. PROBLEM DEFINITION

The number of fast and gradual cores is SoC design detailed and workload dependent. Right here, we focus on a given heterogeneous multi-core processor in every server node, and the situation of taking competencies of these heterogeneous capabilities, primarily compared to utilizing homogenous multi-core processors with the equal vigour price range.
Our purpose is twofold:
1) design a framework for developing virtual Hadoop clusters with different processing capabilities (i.E., clusters with fast and gradual slots); and
2) present a brand new scheduler to support jobs with exclusive performance targets for utilising the created digital clusters and sharing their spare resources. The situation definition is as follows:
enter:
_ C: cluster measurement (quantity of machines)
_ Nf : quantity of fast cores on every computer
_ Ns: quantity of sluggish cores on every computing device
_ S: job size distribution
_ A: job arrival approach

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 6, Special Issue 2, February 2017

National Conference on Recent Advancements in Engineering & Technology and Management 2017 (NCRAETM 2K17)

18th February 2017

Organized by

Sree Rama Engineering College (SRET), Tirupati- 517 507, India

Output: Sched: time table of Map/diminish mission placement
goal: scale downSchedJob Completion Time (Sched)
A traditional first question is why a new Hadoop scheduler is a necessity and why the default Hadoop scheduler can't work well. To answer this question, we show the performance evaluation beneath the identical power finances of making use of the default Hadoop scheduler on heterogenous and homogenous multi-core processors respectively, and in addition our DyScale scheduler with the equal heterogenousmulti-core processors, see determine 1. The important message from figure 1 is that the default Hadoop scheduler can't use good the heterogenous
multi-core processors and can even perform worse than when using it on a cluster with homogenous multicore processors with the same vigor price range as a result of the random use of speedy and gradual cores.

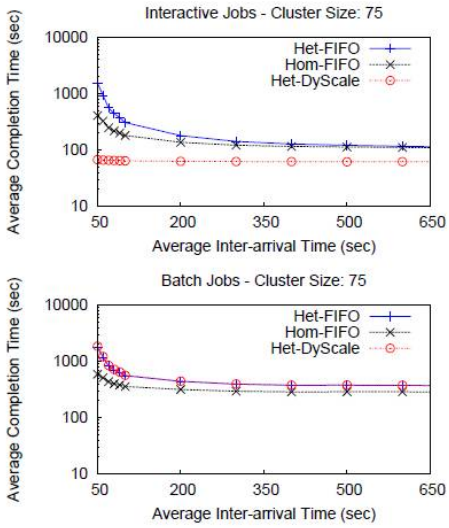


Fig:1The completion time of interactive jobs and batch jobsunder different configurations: heterogenous cluster using FIFO,homogenous cluster using FIFO and heterogenous cluster usingDyScale.

B. DEDICATED VIRTUAL RESOURCE POOLS FOR DIFFERENTJOB QUEUES

DyScale presents the ability to schedule jobs based on efficiency objectives and resource preferences. For illustration, a consumer can put up small, time-sensitive jobs to the Interactive Job Queue to be achieved through fast cores and massive, throughput-oriented jobs to the Batch Job Queue for processing via (many) slow cores. This state of affairs is proven in determine 2. Additionally it is feasible for the scheduler to mechanically respect the job style and time table the job on the suitable queue. For example, small and giant jobs may also be classified headquartered on the number of tasks. A job can be additionally classified headquartered on the appliance information or by including a job style characteristic in job profile. To allocate assets in step with the above scenario, a committed virtual useful resource pool has to be created for every job queue. For example, as proven in figure 2, rapid slots will also be grouped as a digital speedy (vFast) resource pool that is committed to the Interactive Job Queue. Sluggish
slots can be grouped as a virtual slow (vSlow) useful resource pool that is dedicated to the Batch Job Queue.
The TaskTracker asks the JobTracker for a new assignment when the present strolling map/reduce tasks are below the configured highest allowed quantity of map/lessen tasks via a boolean parameter ask- ForNewTask. If the TaskTracker can receive a brand new challenge, then the JobTracker calls the Hadoop Scheduler for a resolution to assign a project to this TaskTracker. The Scheduler tests TaskTrackerStatus to understand whether the on hand slots are Map or curb slots.
DyScales Scheduler also desires to differentiate the slot variety. There are four types of slots: i) rapid map, ii) gradual map, iii) quick scale back, and iv) sluggish slash.
Within the DyScale framework, the Scheduler interacts with the JobQueue by means of since the slot sort, e.G., if the available slot is a quick slot, then this slot belongs to vFast pool, and the Interactive Job Queue is selected for a

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 6, Special Issue 2, February 2017

National Conference on Recent Advancements in Engineering & Technology and Management 2017 (NCRAETM 2K17)

18th February 2017

Organized by

Sree Rama Engineering College (SRET), Tirupati- 517 507, India

job/mission allocation. After opting for the JobQueue, it allocates the on hand slot to the primary job in the queue.

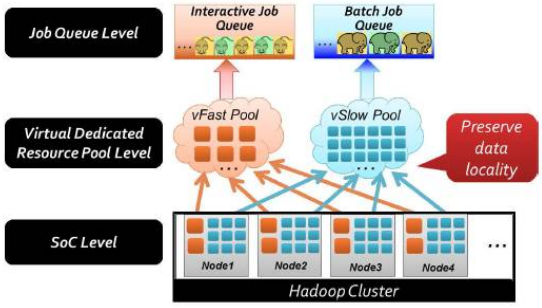


Fig. 2.Virtual Resource Pools.

C. MANAGING SPARE CLUSTER RESOURCES

Static useful resource partitioning and allocation may be inefficient if a resource pool has spare assets (slots) however the corresponding JobQueue is empty, at the same time different JobQueue(s) have jobs that are waiting for resources. For example, if there are jobs in the InteractiveJobQueue and they don't have sufficient fast slots, then these jobs must be able to make use of the to be had (spare) slow slots. We use the virtual Shared (vShare) resource pool to make use of spare assets. As proven in figure three, the spare slots are put into the vShare pool. Slots within the vShare resource pool can be used with the aid of any job queue.

The efficiency of the described resource sharing could be additional accelerated by introducing the TaskMigration mechanism. For instance, the jobs from the Interactive- JobQueue can use spare slow slots until the longer term speedy slots emerge as on hand. These duties are migrated to the newly launched quick slots in order that the roles from the InteractiveJobQueue constantly use most useful resources. Similarly, the migration mechanism permits the batch job to make use of quickly spare quick slots if the InteractiveJobQueue is empty. These resources are lower back through migrating the batch job from the speedy slots to the launched sluggish slots when a brand new interactive job arrives. DyScale permits to specify unique insurance policies for handling spare resources. The migration mechanism is implemented by means of altering the JVMs CPU affinity within the same SoC. With the aid of adding the MIGRATE mission motion in the TaskTrackerAction list in heartbeatResponse, the JobTracker can inform the TaskTacker emigrate the distinct assignment between gradual and quick slots.

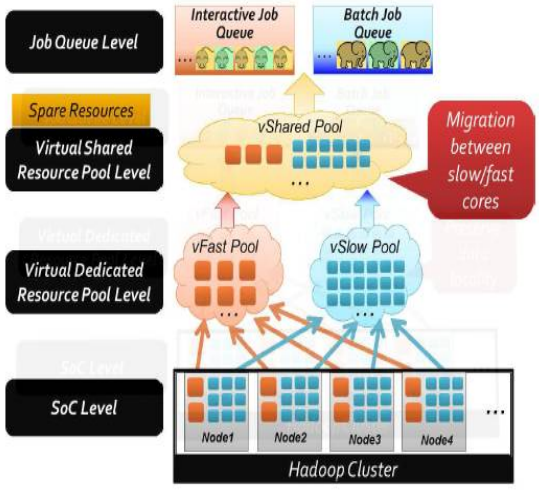


Fig. 3.Virtual Shared Resource Pool.

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 6, Special Issue 2, February 2017

National Conference on Recent Advancements in Engineering & Technology and Management 2017 (NCRAETM 2K17)

18th February 2017

Organized by

Sree Rama Engineering College (SRET), Tirupati- 517 507, India

IV. SIMULATION FRAMEWORK AND RESULTS

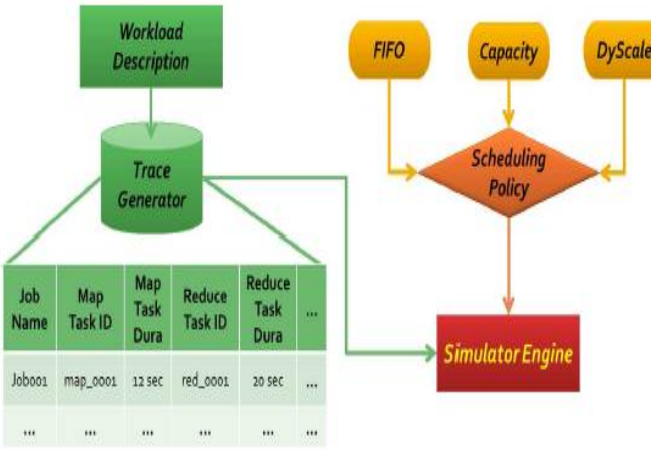


Fig. 4.Simulator Design

As the heterogeneous multi-core processors usually are not but with ease available, we participate in a simulation be trained using the expanded MapReduce simulator SimMR and a synthetic facebook workload. Furthermore, simulation makes it possible for extra comprehensive sensitivity analysis. Our purpose is to compare the job completion times and to participate in a sensitivity evaluation when a workload is accomplished by unique Hadoop clusters deployed on both homogeneous or heterogeneous multi-core processors. The occasion-driven simulator SimMR consists of the following three components, see Figure4:

- A trace Generator creates a replayable MapReduce workload. Additionally, the hint Generator can create traces defined via an artificial workload description that compactly characterizes the period of map and lessen duties as good because the shuffle stage traits by way of corresponding distribution functions. This selection is priceless to behavior sensitivity analysis of latest schedulers and resource allocation insurance policies utilized to distinctive workload varieties.
- The Simulator Engine is a discrete occasion simulator that thoroughly emulates the job master performance within the Hadoop cluster.
- A pluggable scheduling policy dictates the scheduler decisions on job ordering and the amount of assets allotted to different jobs over time.

We prolong SimMR3 to emulate the DyScale framework. We additionally extend SimMR to emulate the capability scheduler for homogeneous environments. We summarize
the three schedulers used on this paper below:

- FIFO: the default Hadoop scheduler that schedules the roles founded on their arrival order.
- capacity: customers can outline exceptional queues for distinct types of jobs. Each and every queue can be configured with a percent of the complete number of slots within the cluster, this parameter is called queue capability. This scheduler has an Elasticity characteristic that allows free resources to be allotted to a queue above its ability to avoid synthetic silos of resources and gain higher resources utilization.
- DyScale: we use two exclusive types: i) the fundamental version without venture migration and ii) the advanced version with the migration function enabled.

With a energy funds of 84W, we select three multicore processor configurations, see desk 1. In our experiments, we simulate the execution of the fb workload on three distinct Hadoop clusters with multi-core processors. For sensitivity evaluation, we gift results for unique cluster sizes of 75, a hundred and twenty, and 210 nodes as they represent intriguing efficiency instances.

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 6, Special Issue 2, February 2017

National Conference on Recent Advancements in Engineering & Technology and Management 2017 (NCRAETM 2K17)

18th February 2017

Organized by

Sree Rama Engineering College (SRET), Tirupati- 517 507, India

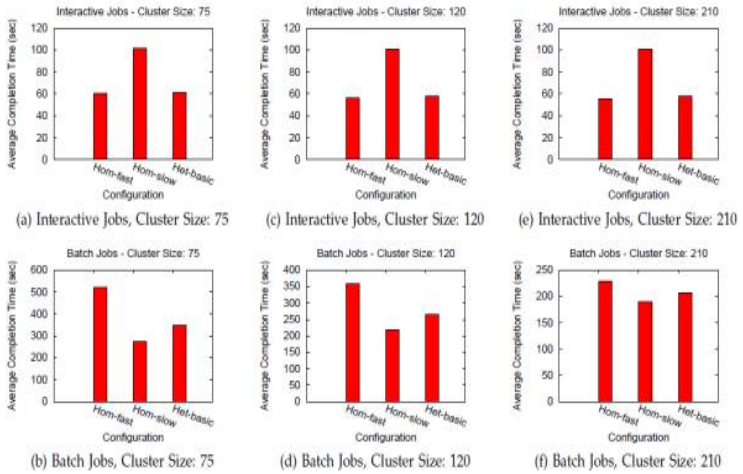
Configuration	Type 1	Type 2	Type 3	Power
Homogeneous-fast	4	0	0	84 W
Homogeneous-slow	0	0	21	84 W
Heterogeneous	0	3	9	84 W

Table 1.Processor configurations with the same power budget of 84 W.

Bin	Map Tasks	Reduce Tasks	# % Jobs
1	1	NA	38%
2	2	NA	16%
3	10	3	14%
4	50	NA	8%
5	100	NA	6%
6	200	50	6%
7	400	NA	4%
8	800	180	4%
9	2400	360	2%
10	4800	NA	2%

Table 2.Job description for each bin in Facebook workload

We generate one thousand MapReduce jobs in keeping with the distribution proven in Table2, with a three-fold broaden in the enter datasets5. Jobs from the first to the 5th team are small interactive jobs (e.G., with lower than 300 duties) and the rest jobs are tremendous batch jobs. The interactive jobs are 82% of the total mix and the batch jobs are 18%. The assignment period of the fb workload will also be nice fit with a LogNormaldistribution and the following parameters: LN(9.9511, 1.6764) for map mission period and LN(12.375, 1.6262) for cut back project period. First, we participate in a evaluation of those three configurations when jobs are processed by way of each cluster in isolation: each and every job is submitted within the FIFO order, there is not any bias due to the targeted ordering policy nor queuing ready time for each job, e.G., each job can use all cluster assets. For the heterogeneous configuration, the SimMR implementation helps the vShared useful resource pool so that a job can use both speedy and gradual resources. Outcome are plotted in figure 5. Each row indicates three graphs that correspond to the clusters with seventy five, 120, and 210 nodes respectively. The graphs exhibit the traditional completion time of interactive jobs (high row) and batch jobs (bottom row).



(a) Interactive Jobs, Cluster Size: 75 (c) Interactive Jobs, Cluster Size: 120 (e) Interactive Jobs, Cluster Size: 210

(b) Batch Jobs, Cluster Size: 75 (d) Batch Jobs, Cluster Size: 120 (f) Batch Jobs, Cluster Size: 210

Fig:5Completion time of interactive and batch jobs under different configurations.

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 6, Special Issue 2, February 2017

National Conference on Recent Advancements in Engineering & Technology and Management 2017 (NCRAETM 2K17)

18th February 2017

Organized by

Sree Rama Engineering College (SRET), Tirupati- 517 507, India

Through evaluating these results, it is obvious that the heterogeneous multi-core processors with quick and gradual cores reward an fascinating design factor. It will possibly drastically improve the completion time of interactive jobs with the equal energy price range. The tremendous batch jobs are benefiting from the greater quantity of the slower cores that toughen throughput of these jobs. Additionally, the batch jobs are able of taking competencies and simply utilizing the extra quick slots in the vShared useful resource pool supported by using DyScale.

V. CONCLUSION

In this work, we make the most the new possibilities and efficiency benefits of utilising servers with heterogeneous multi-core processors for MapReduce processing. Wepresent a new scheduling framework, called DyScale, that's applied on top of Hadoop. DyScale makes it possible for creating exclusive digital resource pools based on the core-types for multi-category job scheduling. This new framework objectives at taking abilities of capabilities of heterogeneous cores for achieving a type of efficiency ambitions. DyScale is convenient to make use of because the created virtual clusters have entry to the equal knowledge stored in the underlying disbursed file approach, and as a result, any job and any dataset will also be processed via either fast or slow virtual useful resource pools, or their mixture. MapReduce jobs may also be submitted into distinctive queues, where they function over exclusive virtual resource pools for attaining higher completion time (e.G.,small jobs) or better throughput (e.G., significant jobs). It is effortless to include the DyScale scheduler into the modern-day Hadoop implementation with YARN , as YARN has a pluggable job scheduler as one of its add-ons.

REFERENCES

[1] T. White, Hadoop:The Definitive Guide. Yahoo Press.
[2] F. Ahmad et al., “Tarazu: Optimizing MapReduce on HeterogeneousClusters,” in Proceedings of ASPLOS, 2012.
[3] J. Dean and S. Ghemawat, “MapReduce: Simplified data processingon large clusters,” Communications of the ACM, vol. 51, no. 1,2008.
[4] M. Zaharia et al., “Delay scheduling: A simple technique forachieving locality and fairness in cluster scheduling,” in Proceedingsof EuroSys, 2010.
[5] Apache, “Capacity Scheduler Guide,” 2010. [Online].Available: http://hadoop.apache.org/common/docs/r0.20.1/capacity scheduler.html
[6] Z. Zhang, L. Cherkasova, and B. T. Loo, “Benchmarking approachfor designing a mapreduce performance model,” in ICPE, 2013,pp. 253–258.
[7] S. Rao et al., “Sailfish: A Framework For Large Scale Data Processing,”in Proceedings of SOCC, 2012.
[8] A. Gates, O. Natkovich, S. Chopra, P. Kamath, S. Narayanam,C. Olston, B. Reed, S. Srinivasan, and U. Srivastava, “Building ahighlevel dataflow system on top of mapreduce: The pigexperience,” PVLDB, vol. 2, no. 2, pp. 1414–1425, 2009.
[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] S. Ren, Y. He, S. Elnikety, and S. McKinley, “Exploiting ProcessorHeterogeneity in Interactive Services,” in Proceedings of ICAC,2013.
[12] H. Esmaeilzadeh, T. Cao, X. Yang, S. M. Blackburn, and K. S.McKinley, “Looking back and looking forward: power, performance,and upheaval,” Commun. ACM, vol. 55, no. 7, 2012.
[13] C. Bienia, S. Kumar, J. Singh, and K. Li, “The PARSEC benchmarksuite:Characterization and architectural implications.” in TechnicalReport TR-811-08, Princeton University, 2008.
[14] “PassMark Software. CPU Benchmarks,” 2013.[Online].Available: http://www.cpubenchmark.net/cpu.php?cpu=Intel+Xeon+E3-1240+%40+3.30GHz
[15] F. Yan, L. Cherkasova, Z. Zhang, and E. Smirni, “Optimizingpower and performance trade-offs of mapreduce job processingwith heterogeneousmulti-core processors,” in Proc. of the IEEE7th International Conference on Cloud Computing (Cloud’2014), June,2014.
[16] A. Verma et al., “Deadline-based workload management formapreduce environments: Pieces of the performance puzzle,” inProc. of IEEE/IFIP NOMS, 2012.
[17] R. Kumar, D. M. Tullsen, P. Ranganathan, N. P. Jouppi, andK. I. Farkas, “Single-isa heterogeneous multi-core architecturesfor multithreaded workload performance,” in ACM SIGARCHComputer Architecture News, vol. 32, no. 2, 2004.
[18] K. Van Craeynest, A. Jaleel, L. Eeckhout, P. Narvaez, and J. Emer,“Scheduling heterogeneous multi-cores through performance impactestimation (pie),” in Proceedings of the 39th InternationalSymposium on Computer Architecture, 2012.
[19] M. Becchi and P. Crowley, “Dynamic thread assignment on heterogeneousmultiprocessor architectures,” in Proceedings of the 3rd conference on Computing frontiers, 2006.
[20] D. Shelepov and A. Fedorova, “Scheduling on heterogeneousmulticore processors using architectural signatures,” in Proceedingsof the Workshop on the Interaction between Operating Systemsand Computer Architecture, 2008.
[21] K. Van Craeynest and L. Eeckhout, “Understanding fundamentaldesign choices in single-isa heterogeneous multicore architectures,”ACM Transactions on Architecture and Code Optimization(TACO), vol. 9, no. 4, p. 32, 2013.
[22] M. Zaharia et al., “Improving mapreduce performance in heterogeneousenvironments,” in Proceedings of OSDI, 2008.
[23] Q. Chen, D. Zhang, M. Guo, Q. Deng, and S. Guo, “Samr: Aself-adaptive mapreduce scheduling algorithm in heterogeneousenvironment,” in IEEE 10th International Conference on Computerand Information Technology (CIT), 2010.

ISSN (Online) : 2319 - 8753
ISSN (Print) : 2347 - 6710

International Journal of Innovative Research in Science, Engineering and Technology

An ISO 3297: 2007 Certified Organization Volume 6, Special Issue 2, February 2017

National Conference on Recent Advancements in Engineering & Technology and Management 2017 (NCRAETM 2K17)

18th February 2017

Organized by

Sree Rama Engineering College (SRET), Tirupati- 517 507, India

[24] R. Gandhi, D. Xie, and Y. C. Hu, "Pikachu: How to rebalanceload in optimizing mapreduce on heterogeneous clusters," inProceedings of 2013 USENIX Annual Technical Conference. USENIX
Association, 2013.
[25] J. Xie et al., "Improving mapreduce performance through dataplacement in heterogeneous hadoop clusters," in Proceedings ofthe IPDPS Workshops: Heterogeneity in Computing, 2010.
[26] G. Gupta, C. Fritz, B. Price, R. Hoover, J. DeKleer, and C. Witteveen,"ThroughputScheduler: Learning to Schedule on HeterogeneousHadoop Clusters," in Proc. of ICAC, 2013.
[27] G. Lee, B.-G. Chun, and R. H. Katz, "Heterogeneity-aware resourceallocation and scheduling in the cloud," in Proceedings of the3rd USENIX Workshop on Hot Topics in Cloud Computing, HotCloud,
2011.
[28] J. Polo et al., "Performance management of accelerated mapreduceworkloads in heterogeneous clusters," in Proceedings of the 41stIntl. Conf. on Parallel Processing, 2010.
[29] W. Jiang and G. Agrawal, "Mate-cg: A map reduce-like frameworkfor accelerating data-intensive computations on heterogeneousclusters," in Parallel Distributed Processing Symposium(IPDPS), 2012 IEEE 26th International, May 2012, pp. 644–655.
[30] Apache, "Apache Hadoop Yarn," 2013. [Online].Available:http://hadoop.apache.org/docs/current/hadoop-yarn/hadoop-yarn-site/YARN.html
[31] A. Verma, L. Cherkasova, and R. H. Campbell, "Resource ProvisioningFramework for MapReduce Jobs with Performance Goals," Proc. of the 12thACM/IFIP/USENIXMiddlewareConference,2011.