www.revistageintec.net
ISSN: 2237-0722
revista GEINTEC Gestão, Inovação e Tecnologias

Adaptive Task Partitioning for Performance Evaluation in Cluster based Heterogeneous Environments

Gosula Anitha[1]; Dr.K. Vengatesan[2]; Dr. Neeraj Sharma[3]

[1]Research Scholar, Department of Computer Science & Engineering, Sri Satya Sai University of Technology & Medical Sciences, Sehore, Bhopal-Indore Road, Madhya Pradesh, India.

[2]Research Guide, Department of Computer Science & Engineering, Sri Satya Sai University of Technology & Medical Sciences, Sehore, Bhopal-Indore Road, Madhya Pradesh, India.

[3]Co-Supervisor, Computer Engineering, Sanjivani College of Engineering, Kopargaon.

Abstract

Due to continual server replacement, datacenter-scale clusters are developing toward heterogeneous hardware designs. Meanwhile, datacenters are frequently used by a variety of users for a variety of purposes. Due to multi-tenant interferences, it frequently exhibits high performance heterogeneity. When contrasted to in-house dedicated clusters, deploying MapReduce on such heterogeneous clusters poses major hurdles in attaining adequate application performance. Heterogeneity can cause significant performance degradation in job execution, despite current optimizations on task scheduling and load balancing, because most MapReduce implementations were developed for homogeneous contexts. To make scheduling decisions, the majority of extant adaptive strategies assume a priori knowledge of particular job characteristics. However, without spending a significant cost, such information is not readily available. The suggested framework Adaptive Control Self-tuning provides a significant improvement over existing methods at moderate to high system utilizations, according to the evaluation results.

Key-words: MapReduce, Cluster, Heterogeneous Systems, Adaptive Task Tuning.

1. Introduction

Over the last few years, MapReduce has shown to be a powerful platform for the treatment of big unstructured data such as system logs, load extracts and web-index calculation. Hardware heterogeneity is caused by the gradual upgrade and substitution of the servers in datacenters. Multiple locals with the same cloud platform can also have varied performance on a standardised hardware. The disparity in MapReduce node handling capabilities can interrupt the assumption that MapReduce

designs homogeneous clusters, resulting in load imbalances, leading to poor performance and minimal use of the cluster. Task scheduling and loaden balancing heterogeneity have been made aware to increase MapReduce performance in heterogeneous situations. Despite these improvements, most MapReduce deployments such as Hadoop are still bad in diverse contexts. MapReduce implementations employ the same setup for jobs to facilitate their management. Research has demonstrated that MapReduce configurations should be determined by the size of the cluster and by the hardware. Therefore, running jobs on heterogeneous nodes with homogenous settings necessarily results to a performance that is not good.

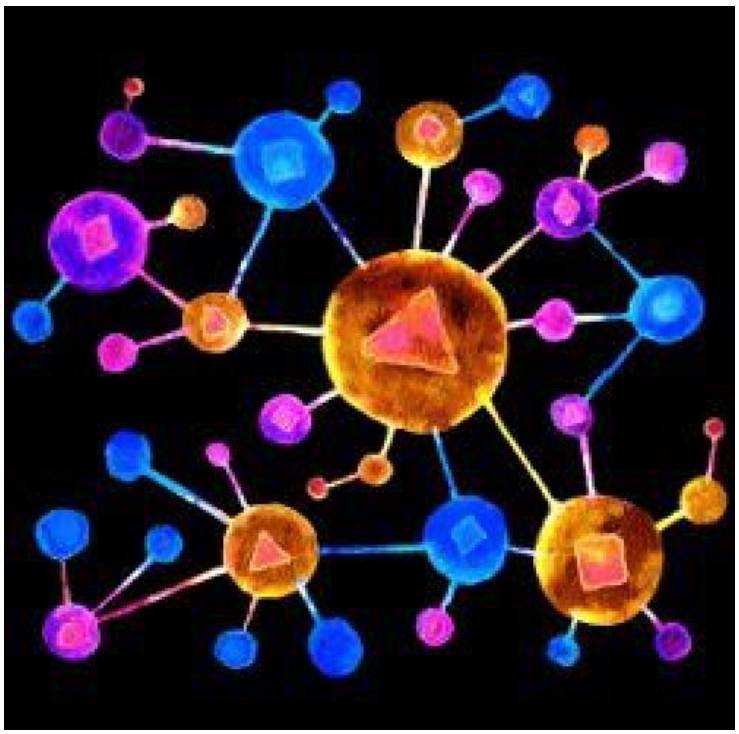
A heterogeneous CPU topology system is a system that uses the same ISA, but the core itself differs in performance. The configuration is more like a symmetric multiprocessor. (While these systems are theoretically asymmetrical multiprocessors, roles and device access do not differ from the core) A common usage of these topologies is to improve mobile soCs' power efficiency. ARM large is the prototype, when rapid high-power cores and slower low-power cores are merged. Similar company Apple Silicon created ARM cores. Intel has also created Lakefield hybrid x86 cores, although the instructions have not been restricted. Connectivity-based clustering, which is also known as Hierarchical Clustering, is based on the primary premise that objects are more closely related than items. These algorithms link "objects" to their distance to "clusters." The greatest distance needed to connect sections of the cluster can be largely stated in a cluster. Different clusters are established at various distances and may be represented by means of a dendrogram explaining the common name of "hierarchical clustering": they do not provide a single partitioning in the data set, but instead provide a wide hierarchy of clusters that merge on certain distances. In a dendrogram, the Y-axis indicates the distance between the clusters, while the objects are positioned along the x-axis to prevent clusters from mixing together.

Connectivity-based clusters are a large series of approaches that differ in their approach of calculating distances. In addition to the standard option of distance functions, the user must also determine which connection criterion to employ (since there are several objects in a cluster, there are several distance calculations). Popular alternatives are called single link clusters (lowest distances), full link clusters (maximum distances of objects), and UPGMA or WPGMA ("Unweighted or Weighted Pair Group Method with Arithmetic Mean", also known as average linkage clustering). In addition, hierarchical clustering (beginning from individual pieces to grouping into clusters) might be agglomerative or separate (starting with the complete data set and dividing it into partitions).

The assessment of clustering findings (or 'validation') is as complex as the clustering process. Popular approaches comprise a "internal" evaluation, where the clustering is summarised as a single

quality point, a " external" assessment comparing the clustering against an existing classification of "ground truth," a "manual" assessment by a human expert and a "indirect" assessment by assessing the usefulness of the clustering in its intended application. Internal evaluation measures are affected by the problem of the role they play as a clustering target itself.

Figure 1 - Heterogeneous Clustering



For example, the data set by the Silhouette factor can be clustered; except that the effective methodology is not known. By employing this internal metric for evaluation, the similarity across optimisation issues is rather compared and the usefulness of clustering not necessarily. External evaluation has similar problems: if we have such "bottom truth" labels, we don't have to cluster them and normally don't have those labels in actual applications. Only one feasible division of the data set reflects the labels, which does not mean that there is no other clustering, and perhaps even better clustering. Therefore, none of these systems can finally determine the true quality of a cluster, but that requires a very subjective human review. Such data may, however, be highly instructive in determining improper clustering, but subjective human judgement should not be dismissed. When an assessment is carried out on the basis of the data itself, an internal assessment is called. These methods usually give the best score to an algorithm, producing clusters with a high resemblance and low resemblance between clusters. The fact that high scores in a cluster assessment do not necessarily

result in excellent information retrieval apps does not have a negative from employing internal criteria. This assessment is also oriented to algorithms using the same cluster concept. K-means, for example, naturally optimise object distances, and the resulting grouping is likely to be exceeded by a distance-based internal criterion.

Performance analysis, usually called profiling, is the examination of behaviour of a programme utilising the data collected during the execution of the programme. Its objective is to decide which programme sections to optimise. A profiler is a performance analysis tool which examines the behaviour and length of function calls in particular. There were tools for performance analysis. Profilers can be characterised by their output kinds or data collection methods. Slow or non-responsive systems can identify a performance problem. This usually happens because high system loading causes a certain element of the system to reach a limited response capacity. Extracting system log information assures that no overhead exists. A detailed timing and frequency adjustment of the queries used for the estimation of Buffer-Miss ratio, table size and number of user processes will not be added to overhead when monitoring the system. This is called a bottleneck within the system A few approaches are employed to boost performance. These include optimization of code, load balancing, caching, distributed computing and self-configuration.

Because heterogeneous computing brings up several new options to build parallel algorithms, our works are driven by further challenges and complexity. One of the issues in order to enhance the system performance is the allocation of tasks (these are task partitions) between the available OpenCL devices. Task partitioning determines how the whole workload is spread (all programme threads) across multiple computer resources. The optimal performance of task splitting is likely to fluctuate with a variety of apps, problem sizes (input) and different hardware setups. Our argument is justified by the presentation of a case study using two programmes that are part of our test cases: linear regression and reduction. The programmes have been run with various problem sizes and task parallels. We measured runtimes in two differentiated target architectures, one CPU and two GPU's.

2. Literature Review

Recent work is important to the newly implemented methodology, as set out below. A hybridised wolf and crow search technique according to the optimum selection of CHs (HGWCSOA-OCHS) has been created and used to increase the life of the network in Subramanian et al. (2020). This was done by a reduction in latency, node distance and energy use. In CH elections, the GWO and CSO hybrid strategy preserves the compromise between exploitation and investigation

in a field of searching. In addition, an unique procedure for finding optimal rendezvous places is called PSO-based selection. (PSOBS). With PSO, the methodology that was established was utilised to identify the ideal meeting sites to get outstanding network resources. Furthermore, a weighted SN value based on the data packets derived from various sensors has been computed.

The research focuses on the identification of a greater number of stragglers, therefore shortening stragglers' long-term execution tail. Nonetheless, a few of these investigations consider the accuracy of the identification of stragglers. In the meanwhile, little work has been done to use reinforcements to reduce the loss of weight From and al (2020). Increased learning represents a class of algorithms for machine learning whereby the agent learns to act in an environment by means of positive and negative rewards. Although we are ever aware of runtime traffickers from the prior job, we prove to be able to advance the accurate identification of traffickers by reinforcing learning before others. Hawkeye can help to reduce working time by double tasks till a phase ends than previous techniques.

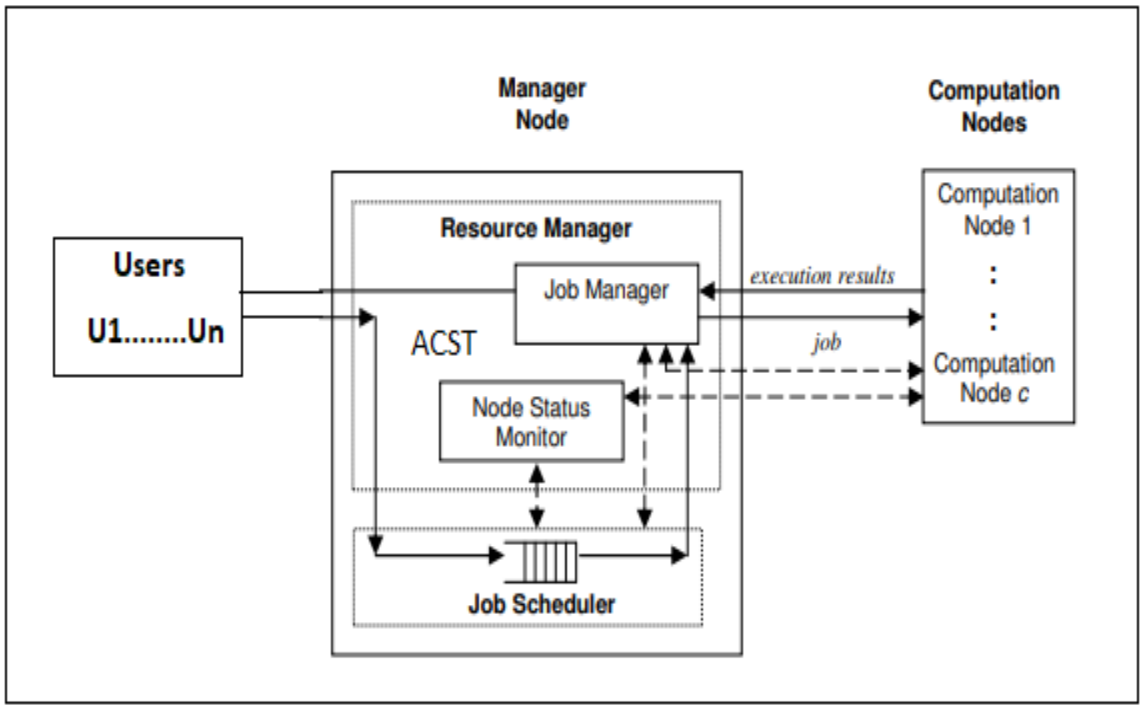
The data-parallel application achieves improved GPU performance. Simultaneously, certain scientific applications are not done adequately on a GPU (i.e. dot product or bread first-search). The same programmes often achieve varied input data sizes in terms of performance. For applications to achieve smaller gains, the input dimension and the type of operations should be based on a different strategy. The load imbalance and sub-optimum execution time for a work pool result in all applications for the GPU. The recognition of the type of application and the processor are therefore important (Sakhnini et al. 2019).

Additional work has addressed the issue of expanding OpenCL platforms to clusters that utilise language extensions to facilitate MPI communication and Rasch et al (2019). It provides a SnuCL framework that allows programmers to see all cluster devices as if they're in the host node. But, in order to move data between nodes, programmers should know about simple MPI calls. Similarly, the libWater library handles communication and data transfers between computer nodes in a transparent form and enables distributed kernel execution on the devices. They built a high level C++ library that simplifies host code development and optimises data transmission and memory management. Whereas these works are very similar to our approach in providing the programmer with an abstraction of the layer, they fundamentally differ from our framework because that clusterCL requires only OpenCL kernels' parameter specification, and kernels and work package partitioning, distribution and scheduling processes are automatically processed within a framework that ensures balance of workloads

3. Research Methodology

This section describes the implementation of Adaptive Control Self-tuning model. Because framework manages resource allocation optimally, i.e. there is no set number of maps and reduction jobs are individually assigned. Consequently the Reduce jobs, which lead to better usage of the capacity available by Map jobs, are not statically programmed by Hadoopsstock (the amount of Reduce jobs are specified in the run-time following the production of the partitions). It employs a two-layer concept for resource scheduling: I the scheduling of jobs by means of resources, (ii) the scheduling of tasks by means of resources. The Resource Manager resource scheduler assigns resources by Application Masters in the first tier and the Application Masters assign container to each job task in the second tier. Focused on the allocation of resources in the second level, the adaptive control. The extent of the study is beyond our remit and we presume that the resource management is fully determining how we may set the proper requirements for each task in the first layer.

Figure 2 - System Architecture



ACST administers resource collection, including processors and cluster disc storage. It retains resource status information to find out which resources are accessible and can therefore assign workloads to available machines. The ACST uses job queues to retain job applications until the resources for executing the jobs are available. The RMS invokes a task planner to choose from the

Received: 10.06.2021 – Accepted: 11.07.2021

queues what jobs to execute when resources are available. The RMS handles workflows and returns the results to the users after completion of work.

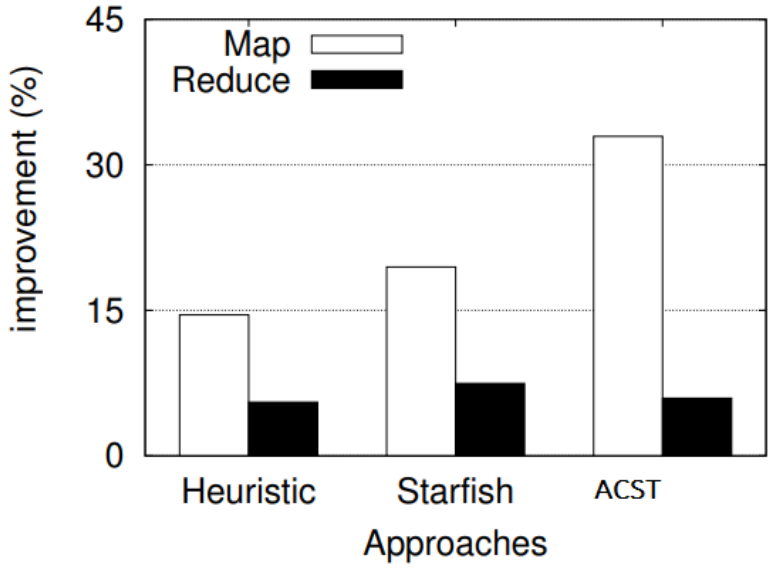
As our past work in heterogeneous task scheduling has shown, adaptive workshops can dramatically minimise overhead in heterogeneous scheduling by anticipating performance rather than using work-queues or discrete jobs. The overhead is lowered if work is statically allocated within the region and the next entry in the region is balanced, lowering the number of jobs to be managed and started. The major negative of our method was, however, that the model only permitted the performance modelling of two devices and prevented them from targeting higher-heterogeneity systems. This paper proposes a solution for the future utilising an integer optimization methodology.

The static schedule for the first time uses our adaptive schedules (Adaptive, Split and Quick). We use the time iteration in the prior pass, which weighted an average of up to five passes, to complete each device as an input in our linear programme for the next pass. All repeated overheads necessary to do an iteration on a specific device are then included (but not one-time overheads such as the copying of persistent data). Therefore, we include and naturally account for data transport and launch costs in the costs of each iteration. In the first occurrence, and then each consecutive instance, the adaptive schedule trains. The divisional timetable divides each region into numerous equally divided sub-regions based on the div entry. Each sub-region is then treated individually and planned for the whole region as Adaptive, which provides a quicker balance of load to the expense of an overhead increase. Split and adaptive schedules are balanced by executing a small sub-region for its first stage of training, which resembles how Split starts. It then schedules all other iterations of the first region instance immediately and uses the adaptive programme in any future case. This plan is suitable for applications that cannot tolerate a whole instance using the static schedule or the overhead of additional scheduling step.

4. Results & Discussion

The improvement comparisons of the ACST, Heuristic and Starfish average completed task time on the physical cluster revealed in Figure 3. It shows that gains in task time are far larger than the reduction of task improvements in all the three configuration techniques. For improving map task performance, ACST considerably beats Heuristic and Starfish while achieving similarly reducing task performance. This is because ACST aims to optimise the intermediate fusion of multi-wave map operations. Most workstations only have a reduction wave, as there are normally 0.9 times the number of reduction workstations available. As a result, map work normally takes place in several

waves whereas reducing jobs for most real-world workloads tends to complete one or two waves. Ant therefore focuses on enhancing map task performance time to utilise the multi-speed characteristic of map tasks. Understanding the wave behaviour of tasks, like the number of waves and the wave size, would help to better the use of the cluster in task settings.



ACST aims to maximise task level performance, especially during the map phase, I/O operations. The task of data leaking plays a major part in the map processing phase and normally takes most of the runtime. In this way Ant seeks to reduce mapping time spills by tuning the parameters associated with the buffer. We quantify the data spilling times for each map task by analysing the spill logs in experiments in order to assess their efficiency.

5. Conclusion

Although it is convenient and easy for use in large-scale parallel and distributed programming for a single design framework such as MapReduce, it ignores the unique needs with diverse platforms and workloads. This study addresses a practical yet tough issue in the automated setup in heterogeneous systems of big MapReduce workloads. We have suggested and developed a tailor-made tuning methodology, Ant, which discovers the ideal settings for particular jobs with heterogeneous nodes automatically. Tasks in ACST are adapted to the heterogeneous nodes with varying settings. It works well for huge operations with several execution rounds of the map. Our experimental results have shown that Ant can increase the average workflow by 25%, 11%, and 16%

Received: 10.06.2021 – Accepted: 11.07.2021

compared to Hadoop's inventory, custom Hadoop with industrial suggestions, and a configuration based profiling approach. We want to extend Ant to a MapReduce multi-tenant system in future development.

References

Subramanian, P., Sahayaraj, J.M., Senthilkumar, S. and Alex, D.S., A Hybrid Grey Wolf and Crow Search Optimization Algorithm-Based Optimal Cluster Head Selection Scheme for Wireless Sensor Networks. *Wireless Personal Communications,* 1-21, 2020

H. Du, S. Zhang, P. Han, K. Zhang, and B. Xu, "Cheetah: A dynamic performance optimization approach on heterogeneous big data analytics cluster," *in Proc. 5th Int. Conf. Big Data Comput. Commun. (BIGCOM),* 2019, 169–177.

Sakhnini J, Karimipour H, Dehghantanha A, Parizi RM, Srivastava G (2019) Security aspects of internet of things aided smart grids: a bibliometric survey. *Internet Things.*

Rasch A, Bigge J, Wrodarczyk M, Schulze R, Gorlatch S (2019) dOCAL: high-level distributed programming with OpenCL and CUDA. *J Super comput.*

Khan NA, Latif MB, Pervaiz N, Baig M, Khatoon H, Baig MZ, Burney A (2019) Smart scheduler for CUDA programming in heterogeneous CPU/GPU environment. *In: The international conference on computer modeling and simulation,* 250–253.

Reddy GT, Reddy MPK, Lakshmanna K, Kaluri R, Rajput DS, Srivastava G, Baker T (2020) Analysis of dimensionality reduction techniques on big data. *IEEE Access 8:* 776–788

Taylor B, Marco VS, Wang Z (2017). Adaptive optimization for OpenCL programs on embedded heterogeneous systems. *In: ACM SIGPLAN/SIGBED conference on languages, compilers, and tools for embedded systems,* pp 11–20

Tsog N, Becker M, Bruhn F, Behnam M, Sjödin M (2019) Static allocation of parallel tasks to improve schedulability in CPU–GPU heterogeneous real time systems. *Annu Conf IEEE Ind Electron Soc, 1:* 4516–4522

Ahmed U, Zafar L, Qayyum F, Islam MA (2018) Irony detector at semeval—2018 task 3: irony detection in English tweets using word graph. *In: The international workshop on semantic evaluation,* 581–586.

K. Chronaki, A. Rico, M. Casas, M. Moretó, R. M. Badia, E. Ayguadé, J. Labarta, and M. Valero. 2017. Task Scheduling Techniques for Asymmetric Multi-Core Systems. *IEEE Transactions on Parallel and Distributed Systems 28,* 7 (2017), 2074–2087. https://doi.org/10.1109/ TPDS.2016.2633347