Global Journal of Computer Science and Technology: B
Cloud and Distributed
Volume 15 Issue 1 Version 1.0 Year 2015
Type: Double Blind Peer Reviewed International Research Journal
Publisher: Global Journals Inc. (USA)
Online ISSN: 0975-4172 & Print ISSN: 0975-4350

A New Efficient Cloud Model for Data Intensive Application

By Rama Satish K V & Dr. N P Kavya

Abstract- Cloud computing play an important role in data intensive application since it provide a consistent performance over time and it provide scalability and good fault tolerant mechanism. Hadoop provide a scalable data intensive map reduce architecture. Hadoop map task are executed on large cluster and consumes lot of energy and resources. Executing these tasks requires lot of resource and energy which are expensive so minimizing the cost and resource is critical for a map reduce application. So here in this paper we propose a new novel efficient cloud structure algorithm for data processing or computation on azure cloud. Here we propose an efficient BSP based dynamic scheduling algorithm for iterative MapReduce for data intensive application on Microsoft azure cloud platform. Our framework can be used on different domain application such as data analysis, medical research, dataminining etc… Here we analyze the performance of our system by using a co-located cashing on the worker role and how it is improving the performance of data intensive application over Hadoop map reduce data intrinsic application. The experimental result shows that our proposed framework properly utilizes cloud infrastructure service (management overheads, bandwith bottleneck) and it is high scalable, fault tolerant and efficient.

Keywords: *big data, scheduling, iterative mapreduce, microsoft azure, hadoop.*

GJCST-B Classification : *C.2.1, E.1*

Strictly as per the compliance and regulations of:

© 2015. Rama Satish K V & Dr. N P Kavya. This is a research/review paper, distributed under the terms of the Creative Commons Attribution-Noncommercial 3.0 Unported License http://creativecommons.org/licenses/by-nc/3.0/), permitting all non-commercial use, distribution, and reproduction inany medium, provided the original work is properly cited.

A New Efficient Cloud Model for Data Intensive Application

Rama Satish K V [α] & Dr. N P Kavya [σ]

Abstract- Cloud computing play an important role in data intensive application since it provide a consistent performance over time and it provide scalability and good fault tolerant mechanism. Hadoop provide a scalable data intensive map reduce architecture. Hadoop map task are executed on large cluster and consumes lot of energy and resources. Executing these tasks requires lot of resource and energy which are expensive so minimizing the cost and resource is critical for a map reduce application. So here in this paper we propose a new novel efficient cloud structure algorithm for data processing or computation on azure cloud. Here we propose an efficient BSP based dynamic scheduling algorithm for iterative MapReduce for data intensive application on Microsoft azure cloud platform. Our framework can be used on different domain application such as data analysis, medical research, dataminining etc… Here we analyze the performance of our system by using a co-located cashing on the worker role and how it is improving the performance of data intensive application over Hadoop map reduce data intrinsic application. The experimental result shows that our proposed framework properly utilizes cloud infrastructure service (management overheads, bandwith bottleneck) and it is high scalable, fault tolerant and efficient.

Keywords: big data, scheduling, iterative mapreduce, microsoft azure, hadoop.

I. Introduction

Cloud computing technology is increasingly getting attention as a future paradigm for hosting, computing and delivering service over the internet. Cloud provides different service which are classified as follows and Infrastructure (Infrastructure as a Service: IaaS), Platform (Platform as a Service: PaaS), Software (Software as a service: SaaS). Cloud service provider provides user to access different type of service such as storage, software or hardware. In particular, in recent years IaaS have become increasingly popular for user to deploy his application on to the cloud for execution and use the cloud resource efficiently. Cloud computing also provides scalable resource computing and storage resources through the Internet [1]–[2]. It also allow cloud users to access services irrespective to where the cloud services are provided and how they are delivered, similar to other essential commodity(electricity, water)[3]. With the adaptable, transparent and scalable features in the service provisioning and resource allocation, more and more data-intensive applications are developed by using cloud computing environment. The data intensive applications spend most of their execution time in disk I/O operation for processing a huge volume of data, e.g. data mining of different enterprises transactions, satellite data processing, medical research computation, etc. Hadoop [4] is a well-known cloud computing platform which is used for data-intensive applications. Due to a large number of nodes in the cloud computing system, the probability of hardware failures is not a big issue based on the statistical analysis of hardware failures in [5]–[6]. Some hardware failures will damage the disk data of nodes. As a result, the running data-intensive applications may not fetch/read map data from disks properly. To come out of these map failures, the data replication technique is used in the cloud computing environment which provides high data availability [7]–[8]. When data map failure occurs, the QoS requirement of the application cannot be supported continuously. The reason is explained as follows. With a large number of nodes in the cloud computing environment, it is practically not possible ask all nodes with the same performance and capacity in their CPUs, memory, and disks [9]. For example, the Amazon EC2 is a well-known heterogeneous cloud platform, which provides various infrastructure resource types to meet different user needs in resource computing and storage [10].

The Microsoft Azure is a cloud computing environment which offers services on demand. It offers on demand computing services such as Windows Azure Compute, Storage Blob, Queue, Table service etc. Azure Compute is a platform as a service infrastructure which allow the users to lease hourly charged virtual machine instances in the form of different types of Roles such as(e.g.: Worker Role, Web Role, etc…). The Azure storage queue is an eventual consistent, reliable, scalable and distributed message queue service, which is ideal for small and transient messages (short period/impermanent). The Azure Storage Blob service provides a shared storage service where users can store and retrieve any type data by using web services interface. Azure Storage Table service provides scalable non-relational highly available structured data storage. However Azure platform currently do not offer a distributed computing framework, other than the simple queue based model.

Goal of our model is to provide and process the efficient execution of Map Reduce and iterative Map Reduce applications in the Azure cloud environment.

Author α σ : e-mails: ramasatish.k.v@rnsit.ac.in, n.p.kavya@rnsit.ac.in

© 2015 Global Journals Inc. (US)

Our model is a distributed decentralized Map Reduce runtime for Windows Azure cloud environment that utilizes Azure infrastructure services. Our model overcomes the latencies of cloud services by using sufficiently fine grained map and reduces tasks. It overcomes the eventual data availability of cloud storage services through re-trying and by explicitly designing the system to not rely on the immediate availability of data across all distributed workers. Our systems uses Azure Queues for map and reduce BSP task scheduling, Azure Tables for metadata storage and for monitoring data storage, Azure Blob storage for data storage (input, output and intermediate) and Compute worker roles to perform the computations. In order to with stand the breakdown/failure of cloud infrastructures and to avoid single point of failures, our model was designed as a decentralized control architecture which does not rely on a client side. It provides users with the capability to scale up/down the number of computing resources such virtual machines dynamically or during runtime. The map and reduce tasks of the proposed model runtime are dynamically scheduled using azure global queues achieving efficient scheduling natural load balancing of tasks. Our system models handles BSP task failures and slower tasks through re-execution and duplications. Map Reduce infrastructure requires the reduce tasks to ensure guarantee of all the intermediate data products from Map tasks before starting the reduce phase.

Data Iterative computation generally relies on a set of static data that remain fixed across different iterations and a set of ephemeral dynamic data between iterations. Here we introduces an in memory co-located Data Cache to store the reusable static data across the iterations, avoiding the fetching and parsing cost of such data from Azure Blob storage for every iteration. Each worker role will have one managed collocated cache with a given memory limit. Since the existing model do not have proper mechanism which can utilize the knowledge about cached data products to assign tasks to workers, scheduling tasks to take advantage of caching presents a significant challenge. At the same time, it's important to maintain the efficient scheduling and fault tolerance of our model in the new scheduling mechanism. In order to address these issues, our model utilizes a new efficient scheduling approach using a combination of Azure Queues and Tables. The first iteration of our model will get scheduled only through Azure queues. Our model uses a special table for caching, where the tasks are retrieved from second iteration onwards. Map Workers first query this table to identify any similarity between the data items they have in their collocated cache vs the data items needed for the retrieval of tasks. With this prototype the static data for data iterative Map Reduce computations will get reused from the second iteration onwards. Meanwhile the newly joined or a worker who has completed processing all the tasks for the cached data will be able to pick up BSP tasks directly from the queue and will use the Azure Tables and the monitoring infrastructure to check the tasks processed or not. This also ensures that our model retains the fault tolerance features of efficient azure cloud structure.

II. Existing System

Over the year cloud computing is growing enormously in various fields. Data intensive application is one of those fields that have been one of the popular topics. In recent research, the valuable knowledge that can be retrieved from petabyte scale datasets is known as Big Data. Using these analyses the researcher can provide better provide better result. This Big Data analysis is used in different domain such data mining, medical research etc… There exists a substantial body of research on resource allocation and scheduling in clouds and data centers that does not consider the resource utilization efficiency (e.g., [12], [13], and [14]). However, here in this literature review, we only discuss briefly the studies that are directly related to resource utilization in data centers. Kaushik and M. Bhandarkar. [11] Proposed a technique to segregate or divide the servers in a HDP cluster into hot zone and cold zones based on their various performance characteristics, where cold zone servers are mostly idling and hot zone are always powered on. Resource utilization/allocation and scheduling in cloud data centers. Mahadik and Hacker [16] proposed scheduling algorithm policy for virtual HPC clusters. They introduced a resource prediction cloud model for each policy to assess the resources required within a cloud, the task queue wait time for requests, and the size of the additional resources required. Palanisamy et al. [15]proposed a Map Reduce cloud model for creating task. Here they create cluster configurations for the Task using Map Reduce to predict and maximize performance based on deadline-perception, allowing the CSP to optimize its resource allotment and reduce the cost. Zaharia et al. [17] have analyzed the problem of (slower node) speculative execution in Map Reduce. Here they developed a simple robust scheduling algorithm called LATE (Longest Approximate Time to End), which uses predicted completion times to execute the job that hurt the response time the most. Lai and Sandholm [18] have developed a system for resources allocation in shared data and compute resource clusters that enhance Map Reduce job scheduling. Their technique is based on keeping apart Map Reduce clusters in VMs with a dynamically modifiable performance. Wang et al. [19] proposed a new job scheduling technique for Map Reduce that improves the overall throughput in job-intensive application without considering the resource consumption. Ren et al. [20] proposed a task scheduling algorithm that boost the completion time of

© 2015 Global Journals Inc. (US)

small Map Reduce jobs. Their system is based on task priorities to make sure the fast response for small tasks. Chang et al. [21] proposed numerous offline and online system for the Map Reduce scheduling complication to minimize the overall task finishing times. Their system is based on resolving a linear program (LP) relaxation. Changjian Wang et al. [22], here they have presented an optimal scheduling algorithm for data map reduce application. Here they have divided algorithm into two stages which are as follows firstly to estimate the node execution time and then to produce proper or optimal task assignment time. Here they only considered map status that is idle or busy for scheduling job to mappers. This approach leads to few problems like long tail and very high scheduling overhead. Qi Chenwe et al. [23]here they provide an analysis of the downfall of existing or recent speculative execution strategies in Map Reduce. Here they present model which affect the overall performance of those technique: jobs that start asynchronously, improper configuration of phase percentage, data skew and abrupt resource competiveness. Based on these terms, here they developed a new speculative execution model called MCP to handle these scenarios. It takes the task cost performance of cluster computing resources into account, aiming at not only reducing the task execution time but also improving the overall cluster throughput. Yang Wang et al. [24] here they have analyzed two general constraints on budget and deadline for the scheduling of a group of Map Reduce tasks as a workflow on a set of vm'sin the cloud. Here first, they focused on the scheduling-length under budget constraints. Then they designed a new algorithm by combining greedy algorithm with dynamic programming techniques for budget allocation on per-stage basis, which was also shown to be balanced. Then, with this result, here they designed two new heuristic algorithms, GGB and GR, which are based on greedy strategies to reduce the time complexity to reduce the scheduling lengths of the workflows without affecting the budget. Our research reveal that both the algorithms exhibiting a unique or significant advantage over the other, are very close to the optimal algorithm in terms of the scheduling time but obtain much lower time overhead. Amrit Pal et al. [25] here they shows the behavior of the hdp cluster with increasing number nodes. The criterion for which the performance is analyzed is the memory parameters. This research will be useful for the developing a hdpcluster. The number of interaction increases as the size of the cluster size increases. If the data size increases and there may be a chance of out of disk then the normal copy script should be used for increasing virtual disks size. Fan Yuanquan et al.[26] here they shows that the existing Map Reduce platform performs poorly on heterogeneous clusters due to skew loads among the reduce jobs. Here they analyze the downfall of current task distribution method in heterogeneous systems. Here they identify two key reasons for the skew loads: and the heterogeneity of worker nodes and the native hash partitioning. Based on these facts they proposed a performance based prediction model which is based on support vector machine called PM-SVM. Here they also proposed a HAP (heterogeneity- aware partitioning) algorithm based on PM-SVM. They implemented the proposed load balance approaches in the HDP. The hadoop load balancer can improve the performance of reduce jobs, and can also improve the resource utilization of hdpclusters.

III. Proposed System

A Map Reduce job divides the input data into individual chunks which are processed by the map jobs in a completely parallel synchronization manner. The output of the maps are fed as the input to the reduce tasks. Thus, the whole framework is involved in scheduling jobs, monitoring them and re-executes the failed jobss.

A cluster is composed of multiple engines. The number of map and reduce tasks is compromised as Map Reduce job which is executed on cluster.Every worker node applies the map function to the local data, and writes the output result to intermediateblob storage. Worker nodes distribute or schedulemap data based on the output keys (produced by the map function), such that all map data belonging to one key is located on the same azure worker node. The worker nodes now process each and every group of output map data, per key, in parallel.

Map Reduce allows for parallel distributed processing of the maps and reduction operation. Provided that each and every mapping operation is self-reliant of the others, all maps can be performed in parallel – though in real scenario this is limited by the number of self-reliant data sources and/or the number of VM's near each source. Similarly, a set of 'reducers' can perform the reduction phase, provided that all data outputs of the map data operation that share the same key are presented to the same reducer at the same time, or we could say that the reduction function is associative. While this method can often appear to be inefficient compared to model that are more sequential, Map Reduce can be applied to significantly larger volume of data than normal servers can handle – a large server farm can use Map Reduce to sort a petabyte of data in only a few hours. The parallelism also provide some possibility of recovering from partial failure of blob storage or server during the operation: if any one mapper or reducer fails, the work can be rescheduled – assuming the input data is still available.

Let us consider a large data application consisting of map and reduce tasks. The Map Reduce job is executed on a cluster. The Map and Reduce functions of Map Reduce are defined with respect to

© 2015 Global Journals Inc. (US)

data pattern/structured in (key, value) pairs. Map takes one pair of data with a type in one data format, and returns a list of pairs in a different format. Then it is processed in the reduce phase by the reduce task with the key-value pair along with the same key. Thus, the reduce phase can only begin only after the map phase ends. This large data application must be completed by deadline$\mathbb{T}$. $\mathbb{X}$ and $\mathbb{Y}$ represents the set of tasks of map and reduce of the application. The set of slots available for executing these map and reduce tasks are indicated by S_1 and S_2 respectively. The resource utilization is symbolized by $\mathbb{R}_{st}$, where s is the slot $\in (S_1, S_2)$ and t is the task $\in (\mathbb{X}, \mathbb{Y})$ executed on the respective slot. The processing time of the task t when executed on the slot s is represented as τ_{st}. The dependencies of the map and reduce tasks are characterized by the variable $\Psi_{vt}, \forall v, t \in (\mathbb{X} \cup \mathbb{Y})$, where Ψ_{vt} will possess the value 1 if t is assigned after the task v else 0.

The main objective is to minimize the resource utilization when executing the Map Reduce application based on the dependencies of reduce tasks on the map tasks. The resource utilization Map Reduce scheduling problem is given as:

$$\sum_{s\in S_1}\sum_{t\in\mathbb{X}} \mathbb{R}_{st}P_{st} + \sum_{s\in S_2}\sum_{t\in\mathbb{Y}}\sum_{v\in(\mathbb{X}\cup\mathbb{Y})} \Psi_{vt}\tau_{st}P_{st} \quad (1)$$

The above equation has to be minimized. Each map task is assigned to a slot for execution. This is given by:

$$\sum_{s\in S_1} P_{st} = 1, \forall t \in \mathbb{X} \quad (2)$$

The each reduce task is assigned to a task which is represented by:

$$\sum_{s\in S_2}\sum_{v\in(\mathbb{X}\cup\mathbb{Y})} \Psi_{vt}, Q_{st} = 1\ \forall t \in (\mathbb{Y}) \quad (3)$$

The processing time of the application should not exceed the deadline. Without exceeding the deadline, the scheduler will assign to the reduce tasks only after finishing the map tasks. This is established as:

$$\sum_{t\in\mathbb{X}} \tau_{st}P_{st} + \sum_{t\in\mathbb{Y}}\sum_{v\in(\mathbb{X}\cup\mathbb{Y})} \Psi_{vt}\tau_{st'}P_{st'} \leq \mathbb{T}, \forall s \in S_1, \forall s' \in S_2 \quad (4)$$

Thus, it is interpreted as:

$$max_{\forall s\in S_1}\sum_{t\in\mathbb{X}} \tau_{st}P_{st} + max_{\forall s'\in S_2}\sum_{t\in\mathbb{Y}} \tau_{st'}Q_{st'} \leq \mathbb{T} \quad (5)$$

As a result, all reducetasks can be assigned after time:

$$max_{\forall s\in S_1}\sum_{t\in\mathbb{X}} \tau_{st}P_{st} \quad (6)$$

The integrity requirements for the decision variables are given by:

$$P_{st} = \{0,1\}, \forall t \in \mathbb{X}, \forall s \in S_1 \quad (7)$$

$$Q_{st} = \{0,1\}, \forall t \in \mathbb{Y}, \forall s \in S_2 \quad (8)$$

The resource utilization solution consists of P and Q^ where,

$$\hat{Q}_{st} = \sum_{v\in(\mathbb{X}\cup\mathbb{Y})} \Psi_{vt}Q_{st}, t \in \mathbb{Y}\ and\ s \in S_2 \quad (9)$$

Here we develop the algorithm for resource utilization which is very efficient for scheduling Map Reduce jobs. The deadline T is specified for the completion of the large data application. However, the user here will specify only the deadline of the job but not the map or reduce phase. Reduce tasks are performed only after the completion of map tasks, thus reduce tasks are completely dependent on the map tasks. Therefore, the data centre should define the deadline for map tasks based on the availability of map slots so that further tasks are carried out by reduce tasks in order to utilize the resource efficiently. Once the map tasks are done with its tasks based on the map slots, the assignments of the reduce tasks are performed based on its reduce slots meeting its deadline. Thus, the design of this proposed algorithm characterizes the resource utilization. Therefore, the resource utilization rate of the slot s is given by:

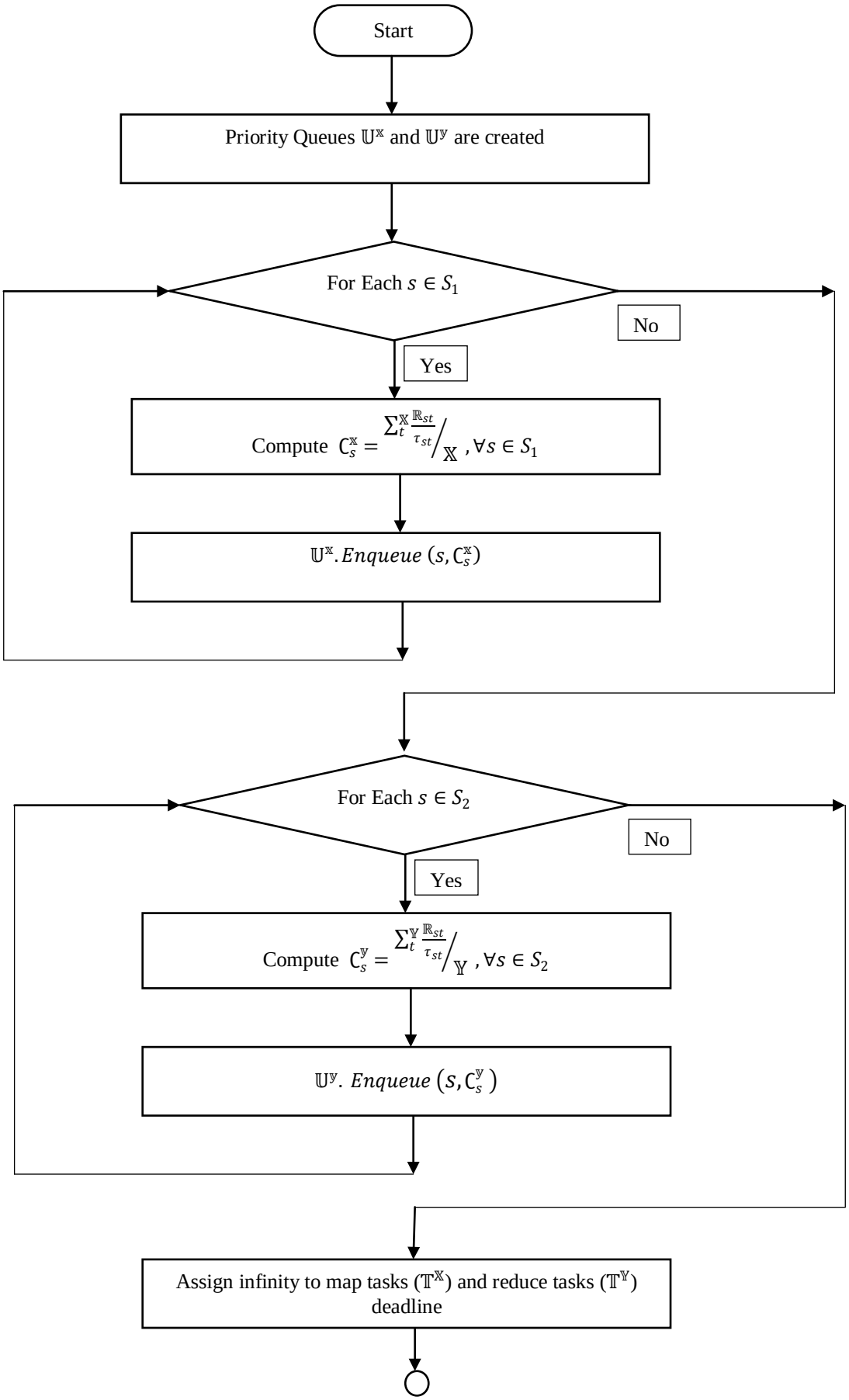
$$C_s^{\mathbb{X}} = {\sum_t^{\mathbb{X}} \frac{\mathbb{R}_{st}}{\tau_{st}}}\Big/{\mathbb{X}}, \forall s \in S_1 \quad (10)$$

$$C_s^{\mathbb{Y}} = {\sum_t^{\mathbb{Y}} \frac{\mathbb{R}_{st}}{\tau_{st}}}\Big/{\mathbb{Y}}, \forall s \in S_2 \quad (11)$$

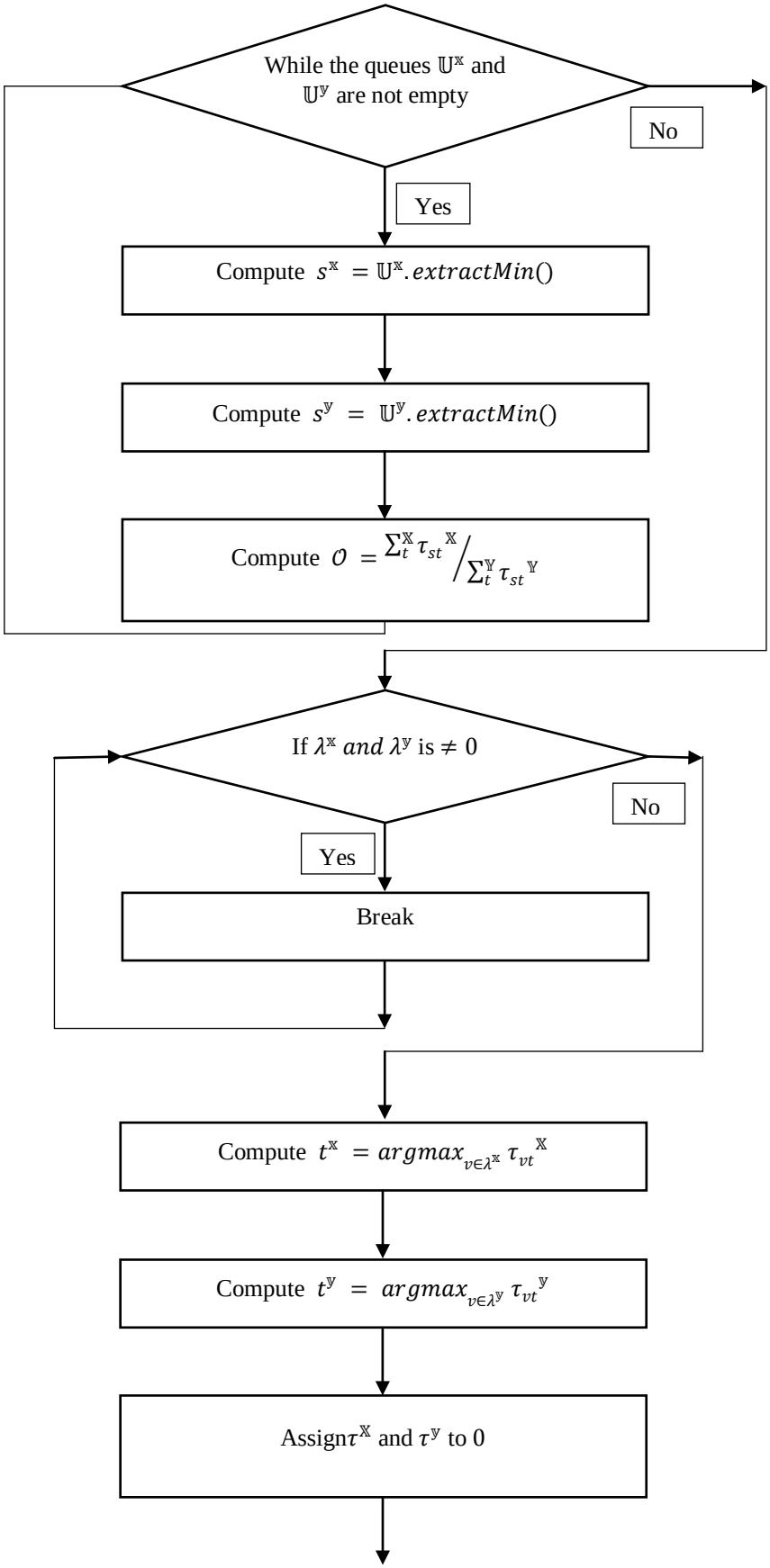
where, it represents the resource utilization rate of map slot s and reduce slot s respectively. The lower $C_s^{\mathbb{X}}$ represents a higher priority for the slot s to which a task is assigned. There exists two priority queues $\mathbb{U}^{\mathbb{X}}, \mathbb{U}^{\mathbb{Y}}$ to keepthe order of the map and reduce slots based on their energyconsumption rates. Our proposed algorithm initializes the deadline for map tasks ($\mathbb{T}^{\mathbb{X}}$) and reduce tasks ($\mathbb{T}^{\mathbb{Y}}$) to infinity.

The complete algorithm is given below.

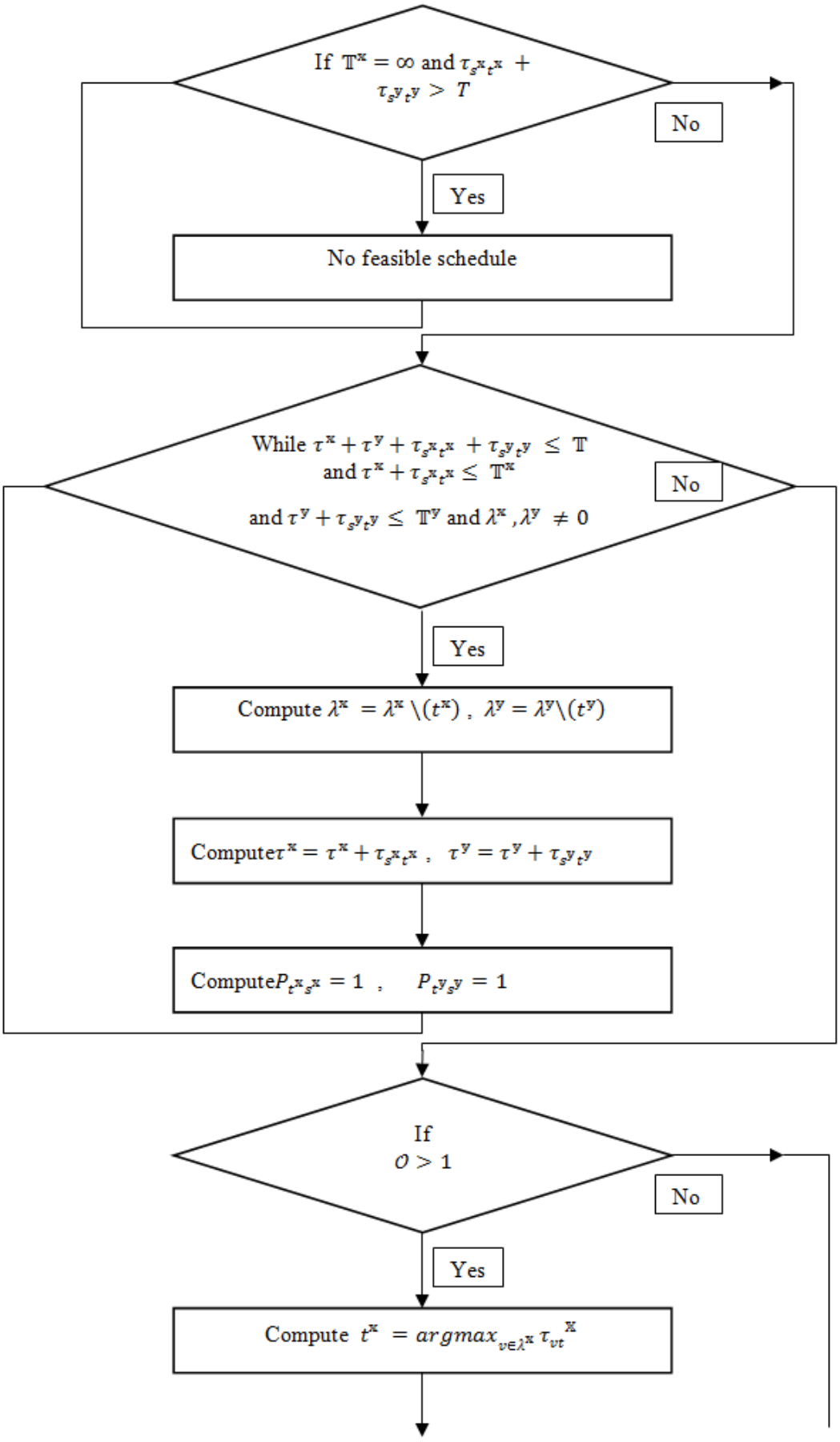
© 2015 Global Journals Inc. (US)



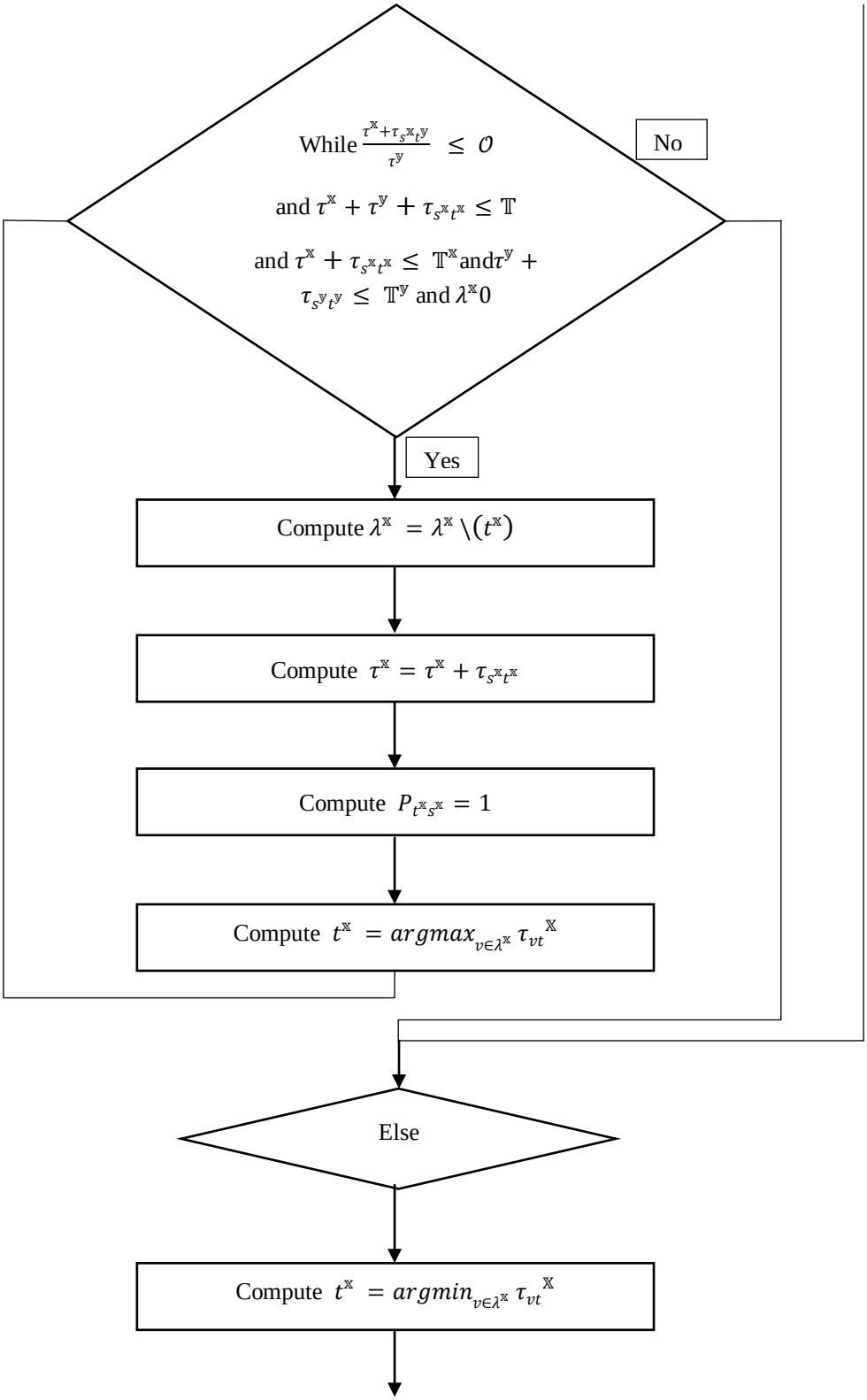
© 2015 Global Journals Inc. (US)



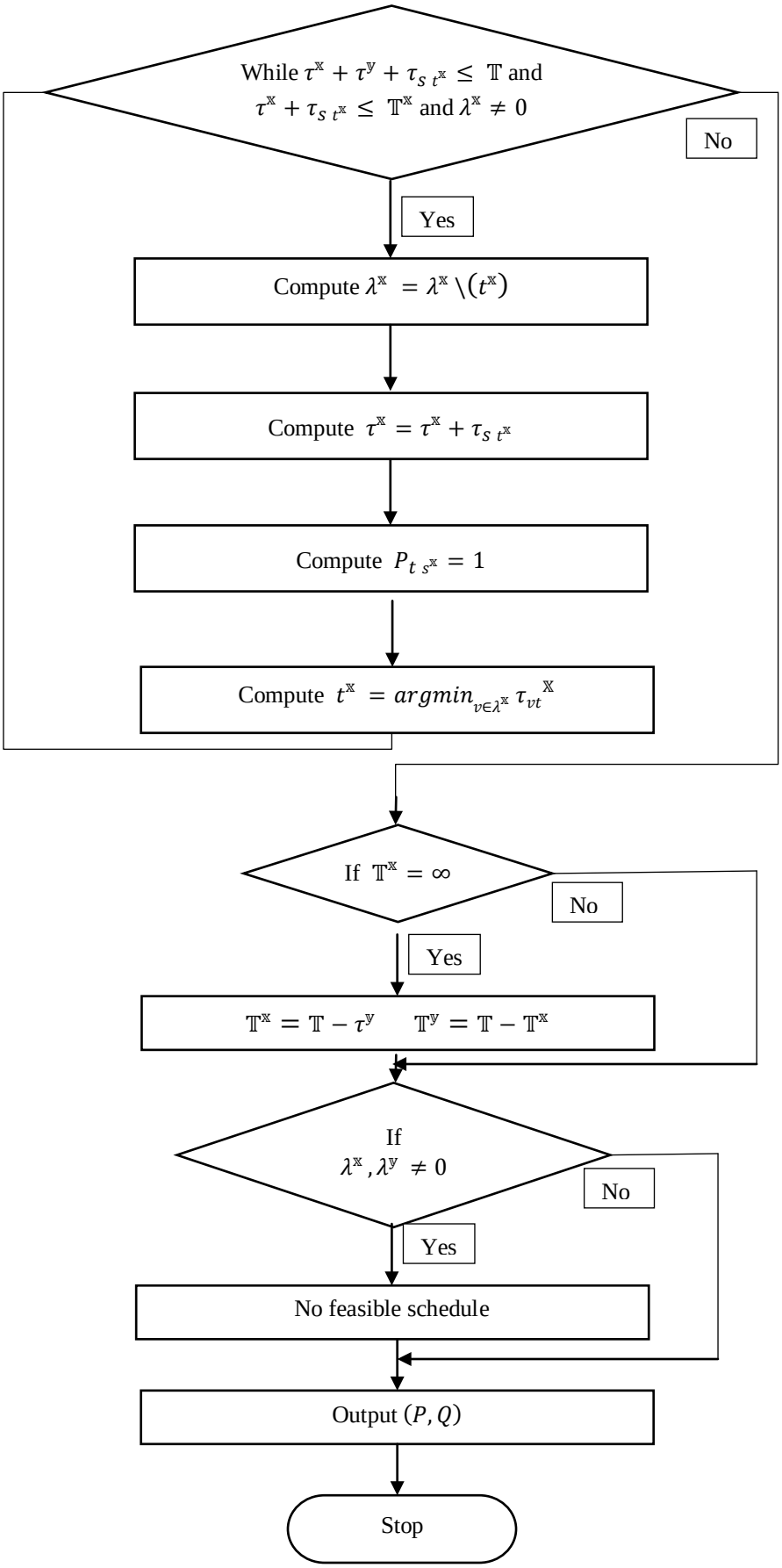
© 2015 Global Journals Inc. (US)



© 2015 Global Journals Inc. (US)



© 2015 Global Journals Inc. (US)



© 2015 Global Journals Inc. (US)

In every iteration, the algorithm chooses the slots with lowest utilization of resource from the priority queues. For these slots, the ratio of processing timeof map tasks to that of the reduce tasks, denoted by $\mathcal{O}$, is calculated. This ratio is used in the task assignmentprocess in each iteration of the algorithm. Then, algorithm sorts the unassigned map and reduce tasks based on theirprocessing time on the selected slots. It selectsthe longest map task$t^{\mathbb{x}}$ and reduce task $t^{\mathbb{y}}$ from the sortedsets $\lambda^{\mathbb{x}}$ and $\lambda^{\mathbb{y}}$ respectively. Then it checksthe feasibility of allocating map task$t^{\mathbb{x}}$ to slot$s^{\mathbb{x}}$ andreduce task $t^{\mathbb{y}}$ to slot $s^{\mathbb{y}}$ by checking the total processingtime of the tasks against the deadline$\mathbb{T}$. If theassignment of map task $t^{\mathbb{x}}$ and reduce task $t^{\mathbb{y}}$ is feasible, the algorithm continues to select tasks from$\lambda^{\mathbb{x}}$and$\lambda^{\mathbb{y}}$, and updates the variables accordingly. To keep the assignments of the tasks in alignment withthe ratio of processing time $\mathcal{O}$, the algorithm balances theassignment. In doing so, if $\mathcal{O} > 1$ (i.e., the load of processingtime of map tasks is greater than that of reduce tasks) andthe ratio of the current assignment is less than $\mathcal{O}$, then thealgorithm assigns more map tasks to balance the allocatedprocessing time close to $\mathcal{O}$. If the ratio of thecurrent assignment is greater than $\mathcal{O}$, the algorithm assignsmore reduce tasks to balance the allocated processing time. After allocating the map and reduce tasks withthe largest processing time, the algorithm assigns small mapand reduce tasks while satisfying the deadline. At the end of the first iteration, the algorithm sets the mapand reduce deadlines based on the allocated tasks. The time complexity of our algorithm is polynomial in thenumber of map slots, the number of reduce slots, the numberof map tasks, and the number of reduce tasks, respectively.

IV. Result

Cloud provide data iterative map reduce as a infrastructure as a service which is modified and executed here. The modified data iterative protocol is used to compute data in azure cloud, which provides the better resources utilization and more importantly provides better scheduling and efficiency. The system model presented has been developed on Visual Studio 2010 framework 4.0 with C#. The overall system has been developed and implemented with Microsoft Azure platform. We have used virtual machine type small with collocated caching. The virtual machine configurations are as follow it uses windows 2008 r2 server, 2.72 GHz with 4 cores with 1.5GB memory.

The developed system has been analyzed for different performance parameters like map resource utilization, Resource utilization based on our proposed model scheme compared with the existing Hadoop model. The relative study for these all factors has been performed. This system or model performance has been verified for various map size, file size with dynamic scheduling as well as performance parameters have been checked for its fault tolerant, robustness justification. The following are the performance analysis of our proposed model over Hadoop.

a) Map Resource Utilization

The map resource utilization of Hadoop and our proposed model is been plotted in the above graph. We have considered a maximum of 8 maps.Here we have taken the execution time by varying the map size and the analytical result proves that the proposed resource utilization time is reduced by35 secfrom 56 secapproximately over Hadoop.

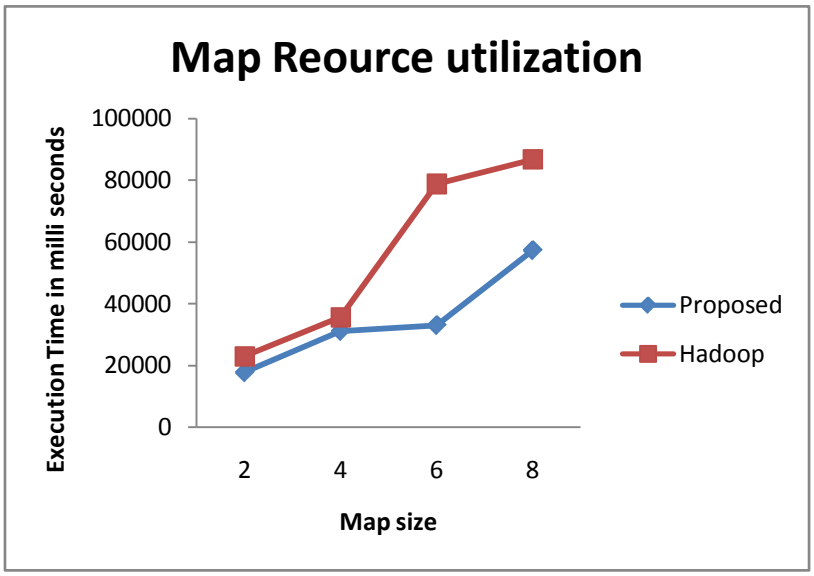


Figure 1 : Map resource utilization

© 2015 Global Journals Inc. (US)

b) Resource utilization

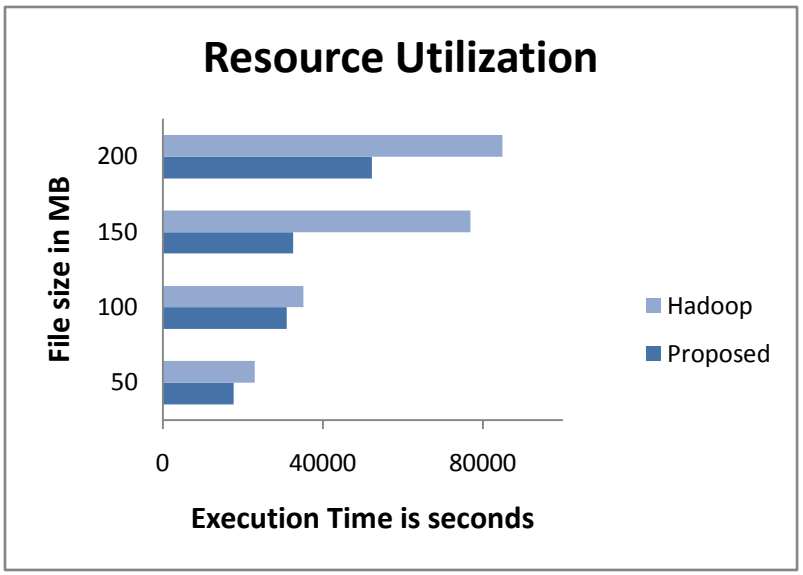


Figure 2 : Map resource utilization

The resource utilization of Hadoop and our proposed model is been plotted in the above graph. We have considered a maximum of 200MB file.Here we have taken the execution time by varying the file size (50, 100, 150, and 200 respectively) and the analytical result proves that the proposed resource utilization time is reduced by33 secfrom 55 secapproximately over Hadoop.

V. Conclusion

Our efficient cloud model provides Map Reduce data intensive computing runtimes for the Microsoft windows azure cloud environment. Our model provides a decentralized iterative expansion to Map Reduce computing environment, enabling the users to easily and efficiently perform task for large scale iterative data analysis/computations on Azure cloud environment. Our model utilizes a BSP scheduling mechanism based on Azure Tables and Queues to provide the caching of static data across iterations in data iterative computations. Our model cloud infrastructure services effectively to deliver robust and efficient applications.

Here we compared the resource utilization and execution time of our proposed model over Hadoop 2.4.0.2.1.3.0-1981. He we analyzed the performance of our model over Hadoop by increasing map size (varying 2, 4, 6, 8 respectively), file size (varying 50Mb, 100Mb, 150Mb, 200Mb respectively) and reduce size by (1, 2, 3, 4) and found that our proposed model is robust and efficient. We also found that by increasing the instance or the number of cores the performance is getting better. We also found how usage collocated caching improves the task execution time.

In future we would like to test this model on different domain type such as data mining, medical research etc… We would also further like to enhance the model by creating a dedicated cache for cache worker which will further improve the performance of our system.

References Références Referencias

1. M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "Above the Clouds: A Berkeley View of Cloud Computing," Dept. EECS, California Univ., Berkeley, Tech. Rep. UCB/EECS-2009-28, Feb. 2009.
2. M. D. Dikaiakos, D. Katsaros, P. Mehra, G. Pallis, and A. Vakali, "Cloud Computing: Distributed Internet Computing for IT and Scientific Research," IEEE Internet Comput. vol. 13, no. 5, pp. 10–13, Sep. 2009.
3. R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud Computing and Emerging IT Platforms: Vision, Hype, and Reality for Delivering Computing as the 5th Utility," Future Gener. Comput. Syst., vol. 25, no. 6, pp. 599–616, Jun. 2009.
4. (2013) Apache Hadoop Project. [Online]. Available: http://hadoop.apache.org
5. K. V. Vishwanath and N. Nagappan, "Characterizing Cloud Computing Hardware Reliability," in Proc. ACM Symp. Cloud Computing, Jun. 2010, pp. 193–204.
6. B. Schroeder and G. A. Gibson, "Disk Failures in the Real World: What Does an MTTF of 1,000,000 Hours Mean to You?" in Proc. 5th USENIX Conf. File and Storage Technologies, Feb. 2007, pp. 1–16.
7. F. Wang, J. Qiu, J. Yang, B. Dong, X. Li, and Y. Li, "Hadoop High Availability through Metadata

© 2015 Global Journals Inc. (US)

Replication," in Proc. 1st Int. Workshop Cloud Data Manage., 2009, pp. 37–44.
8. W. Li, Y. Yang, J. Chen, and D. Yuan, "A Cost-Effective Mechanism for Cloud Data Reliability Management Based on Proactive Replica Checking," in Proc. 2012 12th IEEE/ACM Int. Symp. Cluster, Cloud and Grid Computing (CCGrid), May 2012, pp. 564–571.
9. C. N. Reddy, "A CIM (Common Information Model) Based Management Model for Clouds," in Proc. 2012 IEEE Int. Conf. Cloud Computing in Emerging Markets (CCEM), Oct. 2012, pp. 1–5.
10. (2013) Amazon EC2. [Online]. Available: http://aws.amazon.com/ec2
11. R. T. Kaushik, M. Bhandarkar, and K. Nahrstedt, "Evaluation and analysis of greenhdfs: A self-adaptive, energy-conserving variant of the Hadoop distributed file system," in Proc. of the 2nd IEEE International Conf. on Cloud Computing Technology and Science, 2010, pp. 274–287.
12. B. Moseley, A. Dasgupta, R. Kumar, and T. Sarlos, "On scheduling in map-reduce and flow-shops," in Proc. Of the 23rd Annual ACM Symposium on Parallelism in Algorithms and Architectures, 2011, pp. 289–298.
13. L. Mashayekhy, M. M. Nejad, and D. Grosu, "A truthful approximation mechanism for autonomic virtual machine provisioning and allocation in clouds," in Proc. of the ACM Cloud and Autonomic Computing Conf., 2013, pp. 1–10.
14. M. M. Nejad, L. Mashayekhy, and D. Grosu, "A family of truthful greedy mechanisms for dynamic virtual machine provisioning and allocation in clouds," in Proc. of the 6th IEEE Intl. Conf. on Cloud Computing, 2013, pp. 188–195.
15. B. Palanisamy, A. Singh, and L. Liu, "Cost-effective resource provisioning for mapreduce in a cloud," IEEE Transactions on Parallel and Distributed Systems (forthcoming), 2014.
16. T. J. Hacker and K. Mahadik, "Flexible resource allocation for reliable virtual cluster computing systems," in Proc. ACM Conf. High Perf. Comp., Networking, Storage and Analysis, 2011, p. 48.
17. T. Sandholm and K. Lai, "Mapreduce optimization using regulated dynamic prioritization," in Proc. 11th ACM Int'l Conf. on Measurement and Modeling of Computer Syst., 2009, pp. 299–310.
18. X. Wang, D. Shen, G. Yu, T. Nie, and Y. Kou, "A throughput driven task scheduler for improving mapreduce performance in job-intensive environments," in Proc. of the 2nd IEEE International
19. Congress on Big Data, 2013, pp. 211–218.
20. Z. Ren, X. Xu, M. Zhou, J. Wan, and W. Shi, "Workload analysis, implications and optimization on a production hadoop cluster: A case study on taobao," IEEE Transactions on Services Computing, vol. 7, no. 2, pp. 307–321, 2014.
21. M. Pastorelli, A. Barbuzzi, D. Carra, M. Dell' Amico, and P. Michiardi, "Hfsp: size-based scheduling for Hadoop," in Proc. of IEEE International Conference on Big Data, 2013, pp. 51–59.
22. H. Chang, M. S. Kodialam, R. R. Kompella, T. V. Lakshman, M. Lee, and S. Mukherjee, "Scheduling in mapreduce-like systems for fast completion time," in Proc. of the 30th IEEE International Conference on Computer Communications, 2011, pp. 3074–3082.
23. Changjian Wang; Yuxing Peng; Junyi Liu; Mingxing Tang; Guangming Liu; Jinghua Feng; Pengfei You, "Optimal Task Scheduling in Map Reduce," Networking, Architecture, and Storage (NAS), 2014 9th IEEE International Conference on , vol., no., pp.118,122, 6-8 Aug. 2014
24. Qi Chen; Cheng Liu; Zhen Xiao, "Improving Map Reduce Performance Using Smart Speculative Execution Strategy," Computers, IEEE Transactions on, vol.63, no.4, pp.954,967, April 2014doi: 10.1109/TC.2013.15
25. Yang Wang; Wei Shi, "Budget-Driven Scheduling Algorithms for Batches of Map Reduce Jobs in Heterogeneous Clouds," Cloud Computing, IEEE Transactions on , vol.2, no.3, pp.306,319, July-Sept. 1 2014doi: 10.1109/TCC.2014.2316812
26. Amrit Pal, Sanjay Agrawal, "An Experimental Approach Towards Big Data for Analyzing Memory Utilization on a Hadoop cluster using HDFS and MapReduce".
27. Fan Yuanquan, WU Weiguo, XU Yunlong, CHEN Heng "Improving Map Reduce Performance by Balancing Skewed Loads".

© 2015 Global Journals Inc. (US)

Global Journals Inc. (US) Guidelines Handbook 2015

www.GlobalJournals.org