Volume 05 Issue 01, 2017
ISSN.: 2321-5623

Fault Tolerant Dynamic Task Clustering to Improve Workflow Make Span in Clouds

Vinodhini[1], A. Padmapriya[2]
[1]M.Phil Scholar, Department of Computer Application, Alagappa University, Karaikudi, Tamil Nadu, India.
[2]Associate Professor, Department of Computer Science, Alagappa University, Karaikudi, Tamil Nadu, India.

Abstract: Task clustering is a compute intensive method that will reduce execution overhead there by improving the computational granularity in clouds. Usually a job is composed of one or more tasks. The jobs with multiple tasks are always having higher risk of failures when compared to single task job. Clustering strategies can be used to reduce the impact of job failures. Fault tolerant clustering methods are used to enhance the runtime execution of workflow executions. Static task clustering is the widely employed clustering method for faulty environments. The proposed work utilizes Dynamic task clustering to improve the workflow make span by dynamically modifying the clustering granularity whenever there arises chances of failure. The proposed method performs well to adapt unexpected behaviours and provides better make-spans when compared to the static method.

This paper discusses the implication of the rise of big data and especially that of high velocity data in the domain of high frequency trading (HFT), a growing niche of securities trading. We first take a brief look at the intricacies of HFT including some of the commonly used strategies used by HFT traders. The technological challenges in processing HFT and responding to the real time changes in the market conditions are also discussed. Some of the potential technological solutions to solve the issues thrown up by HFT are analyzed for their effectiveness to address the real time performance requirements of HFT. We identify Complex event processing (CEP) as a candidate to address the HFT problem. The paper is divided into 3 parts; part A deals with understanding HFT and the challenges that it poses to the technological processing. In Part B we look at complex Event Processing (CEP) and the types of problems it can be applied to. In Part C we show a framework to process HFT using techniques derived from CEP.

Keywords: Fault-tolerant, Scientific workflows, Task clustering.

I. Introduction

Cloud computing has recently emerged as a promising hosting platform that permits multiple cloud users referred to as tenants to share a typical physical computing infrastructure. With fast implementation of the thought of software system as a Service (SaaS) and Service Oriented Architecture (SOA), the web has evolved into a very important service delivery infrastructure rather than simply providing host connectivity.

A. Scientific Workflows

A scientific workflow system is a particular type of a workflow management system designed particularly to create and execute a sequence of computational or data handling steps, or workflow, in a scientific application.

B. Fault Tolerance

A fault-tolerant design empowers a framework to proceed with its expected operation, potentially at a decreased level, instead of failing totally, when some piece of the framework fails. The term is most generally used to describe pc systems designed to continue more or less completely operational with, may be, a lessening in throughput or a rise in response time in the event of some halfway failure.

C. Task Clustering

Task clustering method is a simple method used to decrease the execution problems and increase the level of execution.

II. Background Study

In [1] G. Singh, M. Su, K. Vahi, E. Deelamn, B. Berriman, J. Good, D. S. Katz and G. Mehta described about Pegasus Workflow Management System. Examine a different computing model where the user requests a certain number of processors for certain duration from the remote resource and then uses special middleware tools to execute the workflow over the allocated processors without having to go through the remote queue again. This presents advantages for both the user and the resource owner. For the user, he/she has to go through the resource queue only once to get the acquired resources and thus the penalty of queue wait time is only incurred once (provided the workflow make span is less than the maximum wall clock time at the resource). Pegasus Workflow Management System, which maps theoretical Workflow explanation onto distributed computing framework. Pegasus accomplishes dependable, adaptable workflow execution over a wide assortment of computing framework. Techniques used in this system are

Overlaying clustering techniques. The tasks or clusters were submitted to the queue of the NCSA Tera Grid cluster. Since there were multiple levels in the workflows with several tasks or clusters at each level, the effect of the queue wait times get compounded. There are level-based clustering technique can be used. In Level-based clustering, each level of the workflow tasks is clustered based on the label of the task, the tasks in a cluster were executed sequentially. The requested wall clock time of a cluster was the sum of the wall clock times of the tasks in the cluster. So this system have the demerits such as Low latency of job response time and job slow down. In order to execute tasks within a cluster on multiple nodes the process is very difficult for reduced. The result is a reduction in using present technique. the scheduling overhead. Additionally, in the ideal case In [2] W. Chen and E. Deelman without any failures, the clustering described about Task Failure and factor is usually equal to the number Job failure model. The model of all the parallel tasks divided by workflows as Directed Acyclic the number of available resources. Graphs (DAGs), where task portray The gap between DC and SR to clients calculation to be executed indicates that there is still space for and directed edges portray data or further improvement in the control flow conditions between the approach of dynamically adjusting tasks. An unclustered job contains the workflow task clustering factor. only one task that has one process or computation. A clustered job. In [3] G.B. Berriman, G. Juve, contains multiple tasks to be E. Deelman, M. Rynge and J.-S. Vöckler, executed in a sequence or in Vöckler explained about the parallel. The techniques used in this execution of workflow applications system are Task Failure Model and with various I/O, memory and CPU Job Failure Model. In their model, necessities on Amazon EC2, and tasks within a job are executed in a consider the execution of the cloud sequential order. In task clustering, with that of a typical high-the clustering factor is an important performance cluster (HPC). The parameter to influence the objective is to distinguish which performance. The user needs to applications give best execution on define the number of tasks in a the cloud at the least cost. The clustered job. The reason of task technique used in this system is clustering can help enhance the Performance Analysis Clustering. Execution is that it can decrease the Given that Amazon Elastic planning cycles that workflow Computer Cloud uses only process under takings experience commodity hardware and given that since the quantity of tasks has applications make very different demands on resources, it is likely that cost and performance will vary dramatically by application. It was therefore important to study workflow applications that make different demands on resources. Subsequently the objectives of this investigation is to comprehend the execution of three work process applications with different I/O, memory and CPU necessities on a commercial cloud Compare the performance of the cloud with that of a high-performance cluster (HPC) equipped with a high-performance network and parallel file system, and Examine the different expenses related with running workflows on a commercial cloud. Ordering of tasks for execution within a cluster on multiple nodes cannot be implemented.

In [4] N. Muthuvelu, I. Chai, E. Chikkannan, and R. Buyya explained about the increase the subsequent calculation-correspondence proportion by altering the assignment granularity at the grid scheduler. The on-line scheduling algorithm chooses the job granularity established on the dynamic idea of a grid domain: job handling necessities; requirements of network usage conditions; and users QoS requirements. Recreation comes about uncover that our algorithm decreased the general application execution time and communication overhead altogether while fulfilling the runtime constraints set by the users and the resources. The algorithm used in this system is on-line scheduling algorithm. This induced an overhead as the nodes were required to be synchronized after each job group execution iteration. Also limits the flexibility of the job groups by fixing the upper and lower bounds of the number of jobs in the group.

In [5] S. Kalayci, G. Dasgupta, L. Fong, O. Ezenwoye, and S. M. Sadjadi explained about standard Condor DAG Man workflow Large-scale applications, in the form of workflows, may require a planned utilization of sources spreading across multiple administrative domains. Current systems mostly provide a centralized approach to the execution of such workflows across resource domains. This may induce scalability problems for large workflows executing in many domains. In addition, the execution may not be able take advantage of the flexibility of site autonomy.

Large-scale applications, as workflows, may require the planned utilization of assets spreading across different regulatory areas. Adaptable arrangements require a decentralized way to deal with facilitate the execution of such workflows. At runtime, adjustments to the workflow execution plan may be required to meet Quality of Service objectives. They give a decentralized execution way to deal with large-scale workflows on various source domains. It likewise give a low overhead, decentralized runtime adjustment mechanism to enhance the performance of the framework. This method is having the problem of difficulty in the partitioning process and time taken for Execution.

In [6] Y. Zhang and M. S. Squillante described about clustering system focuses on permanent failure rather than ignoring the temporary failure on large scale distributed systems such as cloud and grid. If a particular job fails in multi tasking during clustered job entire job fails even all other same jobs have successfully completed their execution thereby wasting the resources and job cycles. The most common technique used is task clustering is to increase the thickness of workflow tasks executed on distributed resources. If the job fails during runtime, retrying a combined or clustered job is much expensive since all the computed jobs have to be retried and computed again. Grouping and ungrouping of task algorithm is used to control the workflow. Scientific workflows can be made out of many fine computational granularity jobs, where the assignment runtime might be shorter than the framework overhead. Task clustering strategies combines few short jobs into a single task such that the task runtime is expanded and the general framework overhead is reduced. Task clustering is the

most common technique used to address execution overheads and increase the computational granularity of workflow tasks executed on distributed sources.

However, existing clustering methods disparage the effect of failure on the framework.

The Limitations of the Existing System is returning a clustered job can be expensive since completed tasks within the job usually need to be recomputed time taken to retry the failed job is much higher and resource is wasted in higher level while retrying a job. If a particular job fails during multitasking we need to retry all the jobs which is already completed its execution. This method does not concentrate on temporary failure.

III. Fault Tolerant Dynamic Task Clustering

In this proposed work three fault-tolerant task clustering methods to improve existing task clustering techniques in a faulty distributed execution environment. The first method retries failed tasks within a job by extracting them into a new job. The second method dynamically alters the granularity or number of tasks in a job according to the estimated inter-arrival time of task failures. The third method partitions the clustered jobs into finer jobs thereby reducing the job thickness and retries them.

The workflow is submitted to a workflow administration framework that dwells on a client confronting machine called the submit host. This machine can be a client's portable workstation or a group asset. The objective execution condition can be a local machine, like the submit host, a remote physical cluster or grid, or a virtual framework such as the cloud.

The task clustering technique is used to cope with the low performance of short running tasks. These jobs are assembled into coarse-grained jobs to decrease the cost of information exchanges when clustered jobs share input information, and save queuing time when assets are limited. The use of task clustering techniques can significantly improve the workflow execution performance. Where failures are absent, the number of tasks in a clustered job (clustering size, k) would be defined as the number of all tasks in the queue divided by the number of available resources. Such a setting assures that the number of jobs is equal to the number of resources and the workflow can fully utilize the resources. The Task failure detection take that tasks at different levels are often of different types in scientific workflows. Take N independent tasks at the same workflow level and the distribution of the task runtime, the system overheads, and the inter-arrival time of failures, reducing the general runtime M for finishing these jobs by modifying the clustering size k (the quantity of tasks in a job). These task failures are independent for each worker node without considering the failures that bring the whole system down.

The fault tolerant clustering have dynamic re-clustering method which adjust the clustering size (k) of the jobs to reduce the impact of task failures on the workflow execution. Our architecture shows a typical workflow execution environment that targets a homogeneous computer cluster. Then the jobs are executed remotely on individual worker nodes.

Feature of the Proposed Methods are Listed Below

- These clustering methods have been used to improve the fault tolerance of task clustering.
- Resources are not wasted since fault-tolerant task clustering methods are used.
- Extract the largest amount of performance gain.
- Performance of task dynamic-clustering is high.

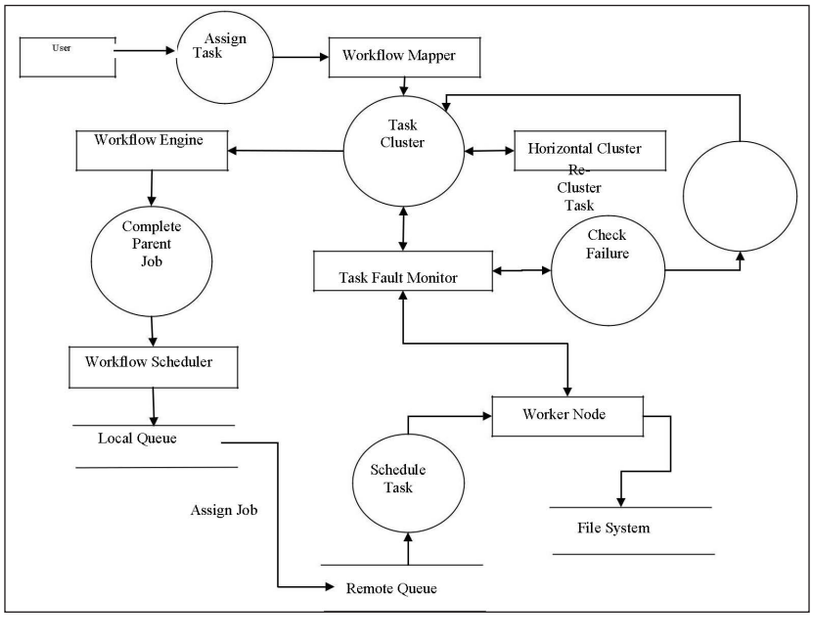

Fig. 1: Dataflow Diagram-Proposed Architecture

IV. CONCLUSION

We modeled transient failures in a distributed environment and assess their influence on task clustering. We proposed three dynamic clustering techniques to enhance the fault tolerance of task clustering. The proposed methods significantly improve the workflow's make span (Time of completion of last job - Starting time of first job) when compared to an existing task clustering method used in workflow management systems. In particular, the dynamic reclustering method. Performed best among all methods since it could adjust the clustering size based on the maximum likelihood estimation of task runtime, system overheads, and the between successive arrival time of failures. The dynamic estimation process, which used data collected during the workflow execution, could further improve the overall runtime in a dynamic environment where the inter-arrival time of failures fluctuated.

References

[1] G. Singh, M. Su, K. Vahi, E. Deelman, B. Berriman, J. Good, D. S. Katz, and G. Mehta, "Workflow task clustering for best effort systems with pegasus," In *Proc. 15th ACM Mardi Gras Conf.,* p. 9, 2008.

[2] W. Chen, and E. Deelman, "Fault tolerant clustering in scientific workflows," In *Proc. IEEE 8th World Congr. Services,* pp. 9-16, 2012.

[3] G. B. Berriman, G. Juve, E. Deelman, M. Rynge, and J.-S. Vöckler, "The application of cloud computing to astronomy: A study of cost and performance," In *Proc. Workshop e-Science Challenges Astron. Astrophys.,* pp. 1-7, 2010.

[4] N. Muthuvelu, I. Chai, E. Chikkannan, and R. Buyya, "On-line task granularity adaptation for dynamic grid applications," In *Proc. 10th Int. Conf. Algorithms Archit. Parallel Process.,* pp. 266-277, 2010.

[5] S. Kalayci, G. Dasgupta, L. Fong, O. Ezenwoye, and S. M. Sadjadi, "Distributed and adaptive execution of condor DAGMan workflows," In *Proc. 22nd Int. Conf. Softw. Eng. Knowl. Eng.,* pp. 587-590, 2010.

[6] Y. Zhang, and M. S. Squillante, "Performance implications of failures in large-scale cluster scheduling," In *Proc. 10th Workshop Job Scheduling Strategies Parallel Process.,* pp. 233-252, June 2004.

[7] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. F. De Rose, and R. Buyya, "CloudSim: A toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Softw.: Practice Experience,* vol. 41, no. 1, pp. 23-50, 2011.

[8] F. Jrad, J. Tao, and A. Streit, "A broker-based framework for multi-cloud workflows," In *Proc. Int. Workshop Multi-Cloud Appl. Federated Clouds,* pp. 61-68, 2013.

[9] Amazon Web Services. [Online]. Available: http://aws.amazon. com, 2014.

[10] FutureGrid. [Online]. Available: http://futuregrid.org, 2014.

[11] Workflow Archive [Online]. Available http://workflowarchive.org , 2014.