

IMPLEMENTATION OF CLOUD COMPUTING SYSTEMS USING MULTI RESOURCE FAIR ALLOCATION

Jenisha S.N¹, D.Dhanya², A. Lenin Fred³

^{1,2} Dept of CSE, Mar Ephraem
College of Engineering and
Technology, Marthandam, Tamil
Nadu, India.

³ Dept of CSE, Asst.Prof, Mar
Ephraem College of Engineering and
Technology, Marthandam, Tamil
Nadu, India.

Email:jenishasn@gmail.com

Abstract- Multi-resource allocation problem is one of the major issues in cloud computing, where the resource pool is constructed from a large number of heterogeneous servers, representing different points in the configuration space of resources such as processing, memory, and storage. A multi-resource allocation mechanism is designed that generalizes the notion of Dominant Resource Fairness (DRF) from a single server to multiple heterogeneous servers. Multi-resource fair allocation provides a number of highly desirable properties. With multi-resource fair allocation, no user prefers the allocation of another user; no one can improve its allocation without decreasing that of the others; and more importantly, no user has an incentive to form a coalition with others to lie about its resource demand. Multi-resource fair allocation also ensures some level of service isolation among the users. As a direct application, a simple heuristic that implements multi-resource fair allocation in real-world systems is designed. Large-scale simulations driven by Google cluster traces show that multi-resource fair allocation significantly outperforms the traditional slot-based scheduler, leading to much higher resource utilization with substantially shorter job completion times.

Keywords- multi-resource allocation, fairness, cloud computing, job scheduling.

1. INTRODUCTION

The cloud provides on-demand network access to a centralized pool of configurable computing resources that can be rapidly provisioned and released with minimal management effort or service provider interaction. Nowadays Cloud computing is rapidly changing the internet service enabling the small organization to build mobile application for users. Cloud computing is a significant advancement in the delivery of information technology and services. Cloud computing builds off a foundation of technologies such as grid computing, which includes clustering, server virtualization and dynamic provisioning, as well as SOA shared services and large-scale management automation. Cloud computing is referred to a model of network computing where program or application runs on a connected servers rather than on a local computing device. Cloud computing relies on sharing of resources to achieve coherence

D. Dolev, D. Feitelson, J. Halpern, R. Kupferman, and N. Linial,[1] a new definition for the simultaneous fair allocation of multiple continuously-divisible resources that we call bottleneck-based fairness (BBF) is used. Roughly speaking, an allocation of resources is considered fair if every user either gets all the resources he wishes for, or else gets at least his entitlement on some bottleneck resource, and therefore cannot complain about not receiving more. CPU usage didn't satisfy.

C. Joe-Wong, S. Sen, T. Lan, and M. Chiang, [2] develops a unifying framework addressing the fairness efficiency trade-off in light of multiple types of resources. It develops two families of fairness functions that provide different trade-offs, characterize the effect of user requests' heterogeneity, and prove conditions under which these fairness measures satisfy the Pareto efficiency, sharing incentive, and envy-free properties. Intuitions behind the analysis are explained in two visualizations of multi resource allocation. More Applications software needs.

C. Reiss, A. Tumanov, G. Ganger, R. Katz, and M. Kozuch,[3]Heterogeneity reduces the effectiveness of traditional slot- and core-based scheduling. Furthermore, some tasks are constrained as to the kind of machine types they can use, increasing the complexity of resource assignment and complicating task migration. The workload is also highly dynamic, varying over time and most workload features, and is driven by many short jobs that demand quick scheduling decisions. Resource Usage is complicated in cloud server.

B. Farley, A. Juels, V. Varadarajan, T. Ristenpart, K. D. Bowers, and M. M. Swf[4] the study of customer-controlled placement gaming is initiated: strategies by which customers exploit performance heterogeneity to lower their costs. It starts with a measurement study of Amazon EC2. It confirms the (oft-reported) performance differences between supposedly identical instances, and leads us to identify fruitful targets for placement gaming, such as CPU, network, and storage performance. It then explores simple heterogeneity-aware placement strategies that seek out better-performing instances. Less security at information retrieval from cloud server.

N. Rajkumar Buyya, Rajiv Ranjan, and Rodrigo Calheiros, [5] CloudSim: an extensible simulation toolkit that enables modeling and simulation of Cloud computing environments is used. The CloudSim toolkit supports modeling and creation of one or more virtual machines (VMs) on a simulated node of a Data Centre, jobs, and their mapping to suitable VMs. It also allows simulation of multiple Data Centers to enable a study on federation and associated policies for migration of VMs for reliability and automatic scaling of applications. Quantifying the performance of resource allocation policies and application scheduling algorithms in Cloud computing environments for different application and service models under varying load, energy performance and system size is a challenging problem.

2 PROPOSED METHODOLOGY

In DRF, the dominant resource is defined for each user as the one that requires the largest fraction of the total availability. The mechanism seeks a maximum allocation that equalizes each user's dominant share, defined as the fraction of the dominant resource the user has been allocated. Suppose that a computing system has 9 CPUs and 18 GB memory, and is shared by two users. User 1 wishes to schedule a set of (divisible) tasks each requiring (1 CPU, 4 GB) and user 2 has a set of (divisible) tasks each requiring (3 CPU; 1 GB). In this example, the dominant resource of user 1 is the memory as each of its task demands $1/9$ of the total CPU and $2/9$ of the total memory. On the other hand, the dominant resource of user 2 is a CPU, as each of its tasks requires $1/3$ of the total CPU and $1/18$ of the total memory. The DRF mechanism, then allocates (3 CPU, 12 GB) to user 1 and (6 CPU, 2 GB) to user 2, where user 1 schedules three tasks and user 2 schedules two. It is easy to verify that both users receive the same dominant share (i.e., $2/3$) and no one can schedule more tasks by allocating additional resources (there is 2 GB memory left unallocated).

The DRF allocation above is based on a simplified all-in-one resource model, where the entire system is modeled as one big server. The allocation hence depends only on the total amount of resources pooled in the system. In the example above, no matter how many servers the system has, and what each server specification is, as long as the system has 9 CPUs and 18 GB memory in total, the DRF allocation will always schedule three tasks for user 1 and two for user 2. However, this allocation may not be possible to implement, especially when the system consists of heterogeneous servers. For example, suppose that the resource pool is provided by two servers. Server 1 has 1 CPU and 14 GB memory, and server 2 has 8 CPUs and 4 GB memory. As shown in figure 1 even allocating both servers exclusively to user 1, at most two tasks can be scheduled, one in each server.

Moreover, even for some server specifications where the DRF allocation is feasible, the mechanism only gives the total amount of resources each user should receive. It remains unclear how many resources a user should be allocated in each server. These problems significantly limit the

application of the DRF mechanism. In general, the allocation is valid only when the system contains a single server or multiple homogeneous servers, which is rarely a case under the prevalent datacenter infrastructure.

.Muti-Resource Fair Allocation And Its Properties:

Instead of allocating separately in each server, DRFH jointly considers resource allocation across all heterogeneous servers. The key intuition is to achieve the max-min fair allocation for the global dominant resources.

Sharing incentive:

Each user should be better off sharing the cluster, than exclusively using her own partition of the cluster. Consider a cluster with identical nodes and n users. Then a user should not be able to allocate more tasks in a cluster partition consisting of $1/n$ of all resources.

Systems design is the process of defining the architecture, components, modules, interfaces, and data for a system to satisfy specified requirements. Systems design could be seen as the application of systems theory to product development. There is some overlap with the disciplines of systems analysis, systems architecture and systems engineering. If the broader topic of product development "blends the perspective of marketing, design, and manufacturing into a single approach to product development," then design is the act of taking the marketing information and creating the design of the product to be manufactured. Systems design is therefore the process of defining and developing systems to satisfy specified requirements of the user.

Object-oriented analysis and design methods are becoming the most widely used methods for computer systems design. The UML has become the standard language in object-oriented analysis and design. It is widely used for modelling software systems and is increasingly used for high designing non-software systems and organizations.

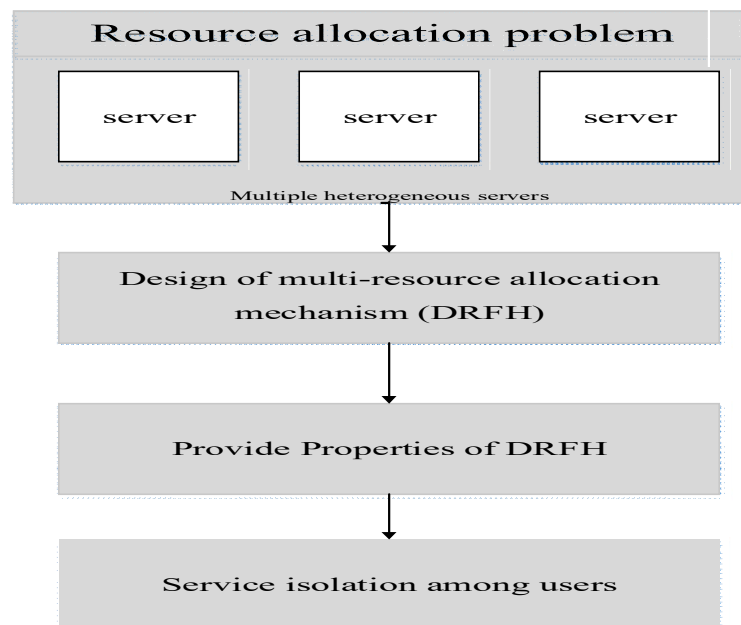


Figure 2.1: Architecture of multi resource allocation

Cloud computing systems represent a much higher diversity in resource demand profiles. Depending on the underlying applications, the workload spanning multiple cloud users may require vastly different amounts of resources (e.g., CPU, memory, and storage) In naive extensions, such as

applying the DRF allocation to each server separately, may lead to a highly inefficient allocation. DRFH generalizes the intuition of DRF by seeking an allocation that equalizes every user's global dominant share, which is the maximum ratio of any resources the user has been allocated in the entire resource pool. Systematically analyze DRFH and show that it retains most of the desirable properties that the all-in-one DRF model provides for a single server.

Recently, there has been a dramatic increase in the popularity of cloud computing systems that rent computing resources on-demand, bill on a pay-as-you-go basis, and multiplex many users on the same physical infrastructure. These cloud computing environments provide an illusion of infinite computing resources to cloud users so that they can increase or decrease their resource consumption rate according to the demands. At the same time, the cloud environment poses a number of challenges. Two players in cloud computing environments, cloud providers and cloud users, pursue different goals; providers want to maximize revenue by achieving high resource utilization, while users want to minimize expenses while meeting their performance requirements

3 PROBLEM DESCRIPTION

In cloud computing system and fairness is a fundamental problem under the notion of resource allocation. Towards addressing the inefficiency of the current allocation module, many recent works focus on multi-resource allocation mechanisms. Notably, Ghodsi et al. [11] suggest a compelling alternative known as the Dominant Resource Fairness (DRF) allocation, in which each user's dominant share—the maximum ratio of any resource that the user has been allocated—is equalized. The DRF allocation possesses a set of highly desirable fairness properties, and has quickly received significant attention in the literature [12]–[15]. While DRF and its subsequent works address the demand heterogeneity of multiple resources, they all limit the discussion to a simplified model where resources are pooled in one place and the entire resource pool is abstracted as one big server. Such an all-in-one resource model not only contrasts the prevalent data centre infrastructure where resources are distributed to a large number of servers but also ignores the server heterogeneity: the allocations depend only on the total amount of resources pooled in the system, irrespective of the underlying resource distribution of servers.

Depending on the underlying server configurations, a computing task may bottleneck on different resources in different servers. In naive extensions, such as applying the DRF allocation to each server separately, may lead to a highly inefficient allocation. This paper represents a rigorous study to propose a solution with provable operational benefits that bridge the gap between the existing multi-resource allocation models and the state-of-the-art data centers infrastructure. In DRFH, a DRF generalization in Heterogeneous environments where resources are pooled by a large number of heterogeneous servers, representing different points in the configuration space of resources such as processing, memory, and storage. DRFH generalizes the intuition of DRF by seeking an allocation that equalizes every user's global dominant share, which is the maximum ratio of any resources the user has been allocated in the entire resource pool.

In analyzing DRFH, it retains most of the desirable properties that the all-in-one DRF model provides for a single server [11]. Specifically, DRFH is Pareto optimal, where no user is able to increase its allocation without decreasing other users' allocations. Meanwhile, DRFH is envy-free in that no user prefers the allocation of another one. More importantly, DRFH is group strategy proof in that no user can schedule more computing tasks by forming a coalition with others to claim more resources that are not needed. The users hence have no incentive to misreport their actual resource demands. In addition, DRFH offers some level of service isolation by ensuring the sharing incentive property in a weak sense it allows users to execute more tasks than those under some "equal partition" where the entire resource pool is evenly allocated among all users. DRFH also satisfies a set of other important properties, namely single-server DRF, single-resource fairness, bottleneck fairness, and population monotonicity.

4 RESULT ANALYSIS

4.1 Screen Shots

4.1.1 Energy node

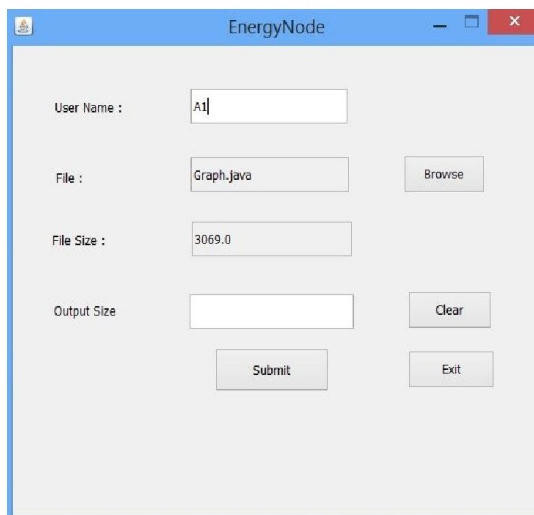


Figure 4.1: Energy node

4.1.2 Energy server node

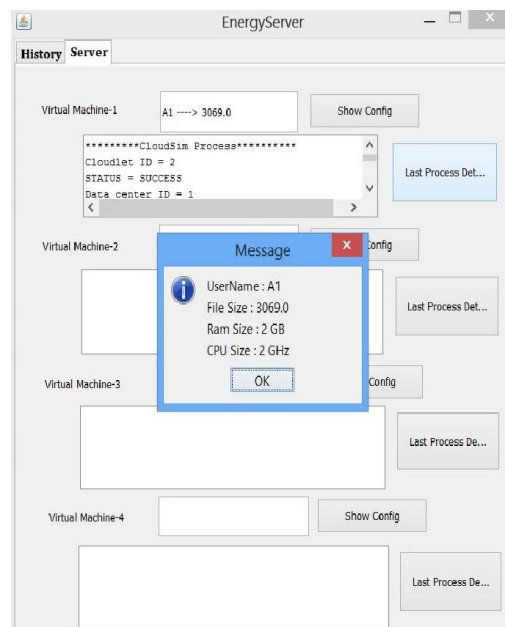


Figure 4.2: Energy server node

4.1.3 Energy server history node



Figure 4.3: Energy history node

4.1.4 Resource Allocation

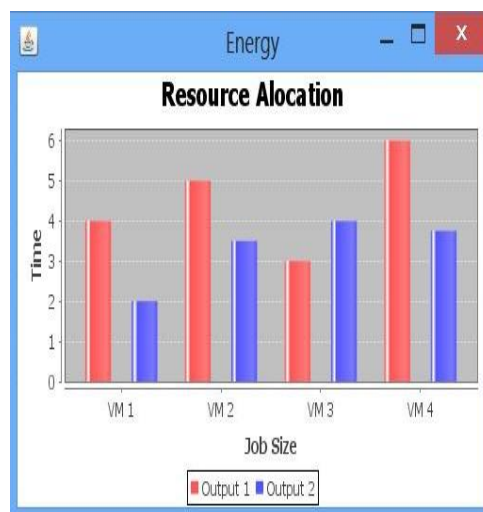
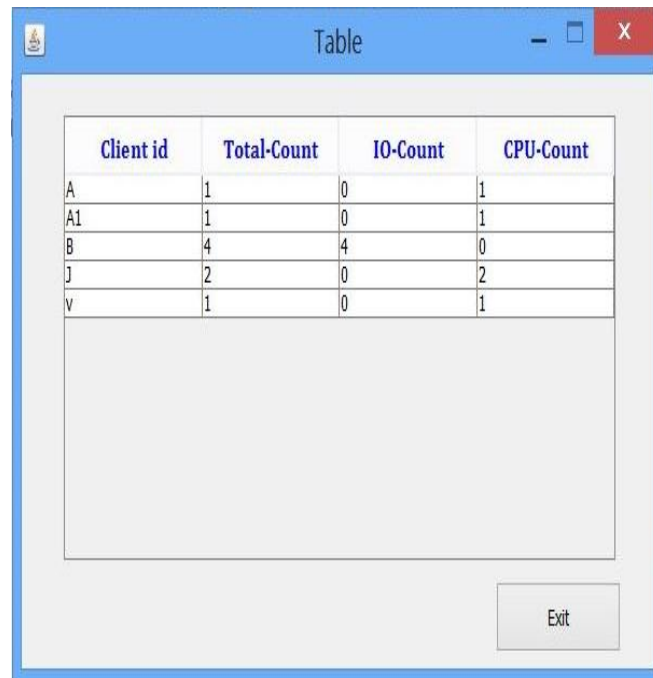


Figure 4.4: Resource Allocation

4.1.5 Table node



Client id	Total-Count	IO-Count	CPU-Count
A	1	0	1
A1	1	0	1
B	4	4	0
J	2	0	2
V	1	0	1

Exit

Figure 4.5: Table node

5 CONCLUSION

A multi-resource allocation problem, a heterogeneous cloud computing system where the resource pool is composed of a large number of servers with different configurations in terms of resources such as processing, memory, and storage. The proposed multi-resource allocation mechanism, known as DRFH, equalizes the global dominant share allocated to each user, and hence generalizes the DRF allocation from a single server to multiple heterogeneous servers. In analyzing multi-resource fair allocation and show that it retains almost all desirable properties that DRF provides in the single-server scenario. Notably, multi-resource fair allocation is envy-free, Pareto optimal, and group strategy proof. It also offers the sharing incentive in a weak sense. Hence design a Best-Fit heuristic that implements DRFH in a real-world system. The large-scale simulations driven by Google cluster traces show that, compared to the traditional single-resource abstraction such as a slot scheduler, DRFH achieves significant improvements in resource utilization, leading to much shorter job completion times.

REFERENCES

- [1] D. Dolev, D. Feitelson, J. Halpern, R. Kupferman, and N. Linial, "No justified complaints: On fair sharing of multiple resources," in Proc. ACM ITCS, 2012.
- [2] C. Joe-Wong, S. Sen, T. Lan, and M. Chiang, "Multi-resource allocation: Fairness-efficiency tradeoffs in a unifying framework," in Proc. IEEE INFOCOM, 2012.

- [3] C. Reiss, A. Tumanov, G. Ganger, R. Katz, and M. Kozuch, "Heterogeneity and dynamicity of clouds at scale: Google trace analysis," in Proc. ACM SoCC, 2012.
- [4] B. Farley, A. Juels, V. Varadarajan, T. Ristenpart, K. D. Bowers, and M. M. Swift, "More for your money: Exploiting performance heterogeneity in public clouds," in Proc. ACM SoCC, 2012.
- [5] N. Rajkumar Buyya, Rajiv Ranjan, and Rodrigo Calheiros, "Modeling and Simulation of Scalable Cloud Computing Environments and the CloudSim Toolkit: Challenges and Opportunities," High Performance Computing & Simulation, 2009.
- [6] Z. Ou, H. Zhuang, A. Lukyanenko, J. Nurminen, P. Hui, V. Mazalov, and A. Yla Jaaski, "Is the same instance type created equal? Exploiting heterogeneity of public clouds," IEEE Trans. Cloud Computing, vol. 1, no. 2, pp. 201–214, 2013.
- [7] F. Ahmad, S. T. Chakradhar, A. Raghunathan, and T. Vijaykumar, "Tarazu: Optimizing mapreduce on heterogeneous clusters," in Proc. ACM ASPLOS, 2012.
- [8] R. Nathuji, C. Isci, and E. Gorbato, "Exploiting platform heterogeneity for power efficient data centers," in Proc. USENIX ICAC, 2007.
- [9] C. Reiss, J. Wilkes, and J. L. Hellerstein, "Google Cluster-Usage Traces," <http://code.google.com/p/googleclusterdata/>.
- [10] "Apache Hadoop," <http://hadoop.apache.org>.
- [11] M. Isard, M. Budiu, Y. Yu, A. Birrell, and D. Fetterly, "Dryad: distributed data-parallel programs from sequential building blocks," in Proc. EuroSys, 2007.
- [12] A. Ghodsi, M. Zaharia, B. Hindman, A. Konwinski, S. Shenker, and I. Stoica, "Dominant resource fairness: Fair allocation of multiple resource types," in Proc. USENIX NSDI, 2011.
- [13] M. Armbrust, A. Fox, R. Griffith, A. Joseph, R. Katz, A. Konwinski, G. Lee, D. Patterson, A. Rabkin, I. Stoica, and M. Zaharia, "A view of cloud computing," Commun. ACM, vol. 53, no. 4, pp. 50–58, 2010.
- [14] W. Wang, B. Li, and B. Liang, "Dominant resource fairness in cloud computing systems with heterogeneous servers," in Proc. IEEE INFOCOM, 2014.
- [15] A. Gutman and N. Nisan, "Fair allocation without trade," in Proc. AAMAS, 2012.