Impact Factor (SJIF): 4.542

International Journal of Advance Research in Engineering, Science & Technology

e-ISSN: 2393-9877, p-ISSN: 2394-2444

Volume 4, Issue 4, April -2017

A Study on Dynamic Resource Allocation and Load Balancing using Gossip Protocol in Large Scale Cloud Environment

Pratijnya Ajawan, Vikrant Shende[2], Subramanya K.N[2]

[1]Dept. Electronics and Communication Engg, Gogte Institute of Technology Belgavi, India
[2] Dept. Electronics and Communication Engg, Gogte Institute of Technology Belgavi, India
[3] Dept. Electronics and Communication Engg, Gogte Institute of Technology Belgavi, India

Abstract: ***The new emerging cloud computing technology is providing various services to users and organizations. Now we are focusing on Infrastructure as a service which allows customers to use resource multiplexing through virtualization technology. Cloud computing applications are developed using Map Reduce programming. Cloud computing distributed file system nodes perform functionality like computing as well as storage in this a file is partitioned into a number of chunks allocated in distinct nodes. based on that Map Reduce tasks can be performed in parallel over the nodes but in a cloud computing environment, failure is the common means nodes may be upgraded, replaced, and added in the system. Files can also be dynamically created, deleted, and appended. This results in load imbalance in a distributed file system; that is, the file chunks are not distributed as uniformly among the nodes. In this project introducing distributed load rebalancing algorithm which eliminated load imbalance problem and proposing gossip protocol which maintains dynamic resource management for large scale cloud. It dynamically maximizing the cloud utility under CPU and memory constraints. In this paper, various resource allocation strategies and their challenges are discussed in detail. It is believed that this paper would benefit both cloud users and researchers in overcoming the challenges faced.***

Keywords- **Cloud resources,** ***Dynamic Resource Allocation,*** **Service Level Agreements (SLA),** ***Resource Scheduling, Load Balancing, Gossip Protocol.***

I. INTRODUCTION

Information concentrated group processing is progressively vital for an expansive number of utilizations including webscale information mining, machine learning, and system activity examination. There has been reestablished enthusiasm for the subject since the production of the MapReduce paper depicting a largescale processing stage utilized at Google. The versatility and the absence of forthright capital venture offered by distributed computing is speaking to numerous organizations. There is a considerable measure of talk on the advantages and expenses of the cloud demonstrate and on the most proficient method to move legacy applications onto the cloud stage. Here we concentrate an alternate issue: in what manner can a cloud specialist co-op best multiplex its virtual assets onto the physical equipment? This is critical in light of the fact that a great part of the touted picks up in the cloud demonstrate originate from such multiplexing. Examines have found that servers in many existing server farms are regularly seriously under-used due to over-provisioning for the pinnacle request. The cloud model is relied upon to make such practice pointless by offering programmed scale up and down in light of load variety. Other than decreasing the equipment cost, it additionally saves money on power which adds to a noteworthy part of the operational costs in extensive server farms. Virtual machine screens (VMMs) like Xen give a component to mapping virtual machines (VMs) to physical assets. This mapping is generally avoided the cloud clients. Clients with the Amazon EC2 benefit for instance, don't know where their VM occurrences run. It is up to the cloud supplier to ensure the hidden physical machines (PMs) have adequate assets to address their issues. VM live movement innovation makes it conceivable to change the mapping amongst VMs and PMs while applications are running. In any case, a strategy issue stays as how to choose the mapping adaptively so that the asset requests of VMs are met while the quantity of PMs utilized is limited. This is testing when the asset needs of VMs are heterogeneous because of the differing set of utilizations they run and shift with time as the workloads develop and shrivel. The limit of PMs can likewise be heterogenous in light of the fact that different eras of equipment exist together in a server farm.

II. RELATED WORK

All works above don't utilize virtual machines and require the applications be organized in a multitier design with load adjusting gave through a front-end dispatcher. Conversely, our work targets Amazon EC2- style condition where it puts no limitation on what also, how applications are developed inside the VMs. A VM is dealt with like a black box. Asset administration is done just at the granularity of entire VMs. MapReduce is another sort of prominent Cloud benefit where information territory is the way to its execution. Quincy embraces min-cost stream display in errand planning to expand information area while keeping decency among various employments. The "Delay Booking" calculation exchanges execution time for information territory. Work allot dynamic needs to employments and clients to encourage asset distribution. VM live movement is a broadly utilized strategy for element asset assignment in a virtualized domain. Our work likewise has a place with this classification. Sandpiper joins multidimensional load data into a solitary Volume metric. It sorts the rundown of PMs in light of their volumes and the VMs in every PM in their volume-to-size proportion (VSR). This shockingly abstracts away basic data required when settling on the movement choice. It then considers the PMs and the VMs in the presorted arrange. Dynamic position of virtual servers to limit SLA infringement. They display it as a container pressing issue and utilize the outstanding first-fit guess calculation to ascertain the VM to PM format intermittently. That calculation, in any case, is outlined for the most part for disconnected utilize. It is probably going to acquire countless when connected in online condition where the asset needs of VMs change progressively.

III. LITERATURE SURVEY

The low average utilization of servers is a well-known cost concern in data center management. Energy costs are rising and low utilization translates into more physical machines, increasing expenditures for machine power and capital and operational costs for cooling systems. Furthermore, excess machines require more floor space and added labor costs. Low utilization has several causes. To guarantee good performance at periods of peak demand, processing capacity is over-provisioned for many business applications. However, processor demand typically exhibits strong daily variability leading to low average utilization. Another source of low utilization is the traditional deployment pattern of one application per OS image and one OS image per unit of physical hardware. This paradigm is typically a consequence of ad-hoc deployment of new applications as it guarantees application isolation and is very easy to implement. Consolidation at the application and OS levels can mitigate inefficiencies in using physical resources. Application consolidation requires considerable skill to ensure isolation between co-hosted applications within an OS image. The usual approach to reducing PC energy wastage is to put computers to sleep when they are idle. However, the presence of the user makes this particularly challenging in a desktop computing environment. Users care about preserving long-running network connections and keeping their machine reachable even while it is idle. Putting a desktop PC to sleep is likely to cause disruption thereby having a negative impact on the user, who might then choose to disable the energy savings mechanism altogether. To reduce user disruption while still allowing machines to sleep, one approach has been to have a proxy on the network for a machine that is asleep. However, this approach suffers from an inherent tradeoff between functionality and complexity because of the need for application-specific customization. In this paper, we present LiteGreen, a system to save desktop energy by employing a novel approach to minimizing user disruption and avoiding the complexity of application-specific customization. The basic idea is to virtualize the user's desktop computing environment, by encapsulating it in a virtual machine (VM), and then migrating it between the user's physical desktop machine and a VM server, depending on whether the desktop computing environment is actively used or idle. When the desktop becomes idle, say when the user steps away for several minutes the desktop VM is migrated to the VM server and the physical desktop machine is put to sleep. When the desktop becomes active again the desktop VM is migrated back to the physical desktop machine.

IV. RESOURCE ALLOCATION STRATEGIES

The input parameters to RAS and the way of resource allocation vary based on the services, infrastructure and the nature of applications which demand resources. The schematic diagram in Fig.1 depicts the classification of Resource Allocation Strategies (RAS) proposed in cloud paradigm. The following section discusses the RAS employed in cloud.

3.1 Execution Time

Different kinds of resource allocation mechanisms are proposed in cloud. In the work by Jiani at.al [8], actual task execution time and preemptable scheduling is considered for resource allocation. It overcomes the problem of resource contention and increases resource utilization by using different modes of renting computing capacities. But estimating the execution time for a job is a hard task for a user and errors are made very often. But the VM model considered in [8] is heterogeneous and proposed for IaaS.

Using the above-mentioned strategy, a resource allocation strategy for distributed environment is proposed by Jose et al. [9]. Proposed matchmaking (assign a resource to a job) strategy in [9] is based on Any-Schedulability criteria for assigning jobs to opaque resources in heterogeneous environment. This work does not use detailed knowledge of the scheduling policies used at resources and subjected to AR's (Advance Reservation).

3.2 Policy

Since centralized user and resource management lacks in scalable management of users, resources and organization-level security policy, Dongwan et al. [10] has proposed a decentralized user and virtualized resource management for IaaS by adding a new layer called domain in between the user and the virtualized resources. Based on role based access control (RBAC), virtualized resources are allocated to users through domain layer.
One of the resource allocation challenges of resource fragmentation in multi-cluster environment is controlled by the work given by Kuo-Chan et al. [11], which used the most-fit processor policy for resource allocation. The most-fit policy allocates a job to the cluster, which produces a leftover processor distribution, leading to the most number of immediate subsequent job allocations.
It requires a complex searching process, involving simulated allocation activities, to determine the target cluster. The clusters are assumed to be homogeneous and geographically distributed. The number of processors in each cluster is binary compatible. Job migration is required when load sharing activities occur.
Experimental results show that the most-fit policy has higher time complexities but the time overheads are negligible compared to the system long time operation. This policy is practical to use in a real system.

3.3 Virtual Machine (VM)

A system which can automatically scale its infrastructure resources is designed in [12]. The system composed of a virtual network of virtual machines capable of live migration across multi- domain physical infrastructure. By using dynamic availability of infrastructure resources and dynamic application demand, a virtual computation environment is able to automatically relocate itself across the infrastructure and scale its resources. But the above work considers only the non-preemptable scheduling policy. Several researchers have developed efficient resource allocations for real time tasks on multiprocessor system. But the studies, scheduled tasks on fixed number of processors. Hence it is lacks in scalability feature of cloud computing [13]. Recent studies on allocating cloud VMs for real time tasks focus on different aspects like infrastructures to enable real-time tasks on VMs and selection of VMs for power management in the data centre. But the work by Karthik et al. [13], have allocated the resources based on the speed and cost of different VMs in IaaS. It differs from other related works, by allowing the user to select VMs and reduces cost for the user.

Users can set up and boot the required resources and they have to pay only for the required resources. It is implemented by enabling the users to dynamically add and/or delete one or more instances of the resources on the basis of VM load and the conditions specified by the user. The above-mentioned RAS on IaaS differs from RAS on SaaS in cloud because SaaS delivers only the application to the cloud user over the internet.

Zhen Kong et al. have discussed mechanism design to allocate virtualized resources among selfish VMs in a non-cooperative cloud environment in [14]. By non-cooperative means, VMs care essentially about their own benefits

without any consideration for others. They have utilized stochastic approximation approach to model and analyze QoS performance under various virtual resource allocations. The proposed stochastic resource allocation and management approaches enforced the VMs to report their types truthfully and the virtual resources can be allocated efficiently. The proposed method is very complex and it is not implemented in a practical virtualization cloud system with real workload.

3.4 Gossip

We address the issue of asset administration for an extensive scale cloud condition that hosts locales. Our commitment bases on sketching out dispersed middleware engineering and displaying one of its key components, a babble convention that meets our outline objectives: decency of asset designation regarding facilitated destinations, productive adjustment to load changes and adaptability as far as both the quantity of machines and locales. We formalize the asset portion issue as that of powerfully amplifying the cloud utility under CPU and memory limitations. While we can demonstrate that an ideal arrangement without considering memory imperatives is direct (yet not valuable), we give an effective heuristic answer for the total issue. We assess the convention through recreation and observe its execution to be very much adjusted to our outline objectives. We consider the issue of asset administration for a huge scale cloud condition. Such a domain incorporates the physical foundation and related control usefulness that empowers the provisioning and administration of cloud administrations. The point of view we take is that of a cloud specialist organization, which has locales in a cloud situation. This work contributes towards building a middleware layer that performs asset assignment in such a cloud domain, with the accompanying plan objectives.

1) Performance objective: We consider computational and memory assets and the goal is to accomplish max-min reasonableness for computational assets under memory limitations.

2) Adaptability: The asset portion prepare should progressively and effectively adjust to changes in the interest for cloud administrations.

3) Scalability: The Resource assignment prepare must be versatile both in the quantity of machines in the cloud and the quantity of destinations that the cloud has. This implies the assets expended per machine keeping in mind the end goal to accomplish a given execution objective must expand sub directly with both the quantity of machines and the quantity of locales. The parts of the middleware layer keep running on all machines. The assets of the cloud are fundamentally devoured by module cases whereby the usefulness of a site is comprised of at least one modules. In the middleware, a module either contains some portion of the administration rationale of a site or a site chief . It has two parts: a request profiler and a demand forwarder. The request profiler gauges the asset request of every module of the site in view of demand measurements. This request gauge is sent to all machine chiefs that run cases of modules having a place with this site. Ask for sending choices consider the asset allotment strategy and limitations, for example, session fondness**.**

3.5. Utility Function

There are many proposals that dynamically manage VMs in IaaS by optimizing some objective function such as minimizing cost function, cost performance function and meeting QoS objectives. The objective function is defined as Utility property which is selected based on measures of response time, number of QoS, targets met and profit etc. There are few works [17] that dynamically allocate CPU resources to meet QoS objectives by first allocating requests to high priority applications. The authors of the papers do not try to maximize the objectives. Hence the authors' Dorian et al. proposed Utility (profit) based resource allocation for VMs which use live VM migration (one physical machine to other) as a resource allocation mechanism [18]. This controls the cost-performance trade-off by changing VM utilities or node costs. This work mainly focuses on scaling CPU resources in IaaS. A few works are also there that use live migration as a resource provisioning mechanism but all of them use policy based heuristic algorithm to live migrate VM which is difficult in the presence of conflicting goals.

For multitier cloud computing systems (heterogeneous servers), resource allocation based on response time as a measure of utility function is proposed by considering CPU, memory and communication resources in [19]. HadiGoudarzi et al. characterized the servers based on their capacity of processing powers, memory usage and communication bandwidth.

For each tier, requests of the application are distributed among some of the available servers. Each available server is assigned to exactly one of these applications tiers i.e. server can only serve the requests on that specified server. Each client request is dispatched to the server using queuing theory and this system meets the requirement of SLA such as response time and utility function based on its response time. It follows the heuristics called force-directed resource management for resource consolidation. But this system is acceptable only as long as the client behaviours remain stationary. But the work proposed in [20] considers the utility function as a measure of application satisfaction for specific resource allocation (CPU, RAM). The system of data centre with single cluster is considered in [20] that support heterogeneous applications and workloads including both enterprise online applications and CPU-intensive applications. The utility goal is computed by Local Decision Module (LDM) by taking current work load of the system. The LDMs interact with Global Decision Module (GDM) and that is the decision-making entity within the autonomic control loop. This system relies on a two-tier architecture and resource arbitration process that can be controlled through each application's weight and other factors.

3.6 Hardware Resource Dependency

In paper [21], to improve the hardware utilization, Multiple Job Optimization (MJO) scheduler is proposed. Jobs could be classified by hardware-resource dependency such as CPU-bound, Network I/O-bound, Disk I/O bound and memory bound. MJO scheduler can detect the type of jobs and parallel jobs of different categories. Based on the categories, resources are allocated. This system focuses only on CPU and I/O resource.

Eucalyptus, Open Nebula and Nimbus are typical open source frame works for resource virtualization management [22]. The common feature of these frameworks is to allocate virtual resources based on the available physical resources, expecting to form a virtualization resource pool decoupled with physical infrastructure. Because of the complexity of virtualization technology, all these frameworks cannot support all the application modes. The system called Vega Ling Cloud proposed in paper [22] supports both virtual and physical resources leasing from a single point to support heterogeneous application modes on shared infrastructure.

Cloud infrastructure refers to the physical and organizational structure needed for the operation of cloud. Many recent researches address the resource allocation strategies for different cloud environment. Xiaoping Wang et al. have discussed adaptive resource co-allocation approach based on CPU consumption amount in [23]. The stepwise resource co-allocation is done in three phases. The first phase determines the co-allocation scheme by considering the CPU consumption amount for each physical machine (PM). The second phase determines whether to put applications on PM or not by using simulated annealing algorithm which tries to perturb the configuration solution by randomly changing one element. During phase 3, the exact CPU share that each VM occupies is determined and it is optimized by the gradient climbing approach. This system mainly focuses on CPU and memory resources for co-allocation and does not considered the dynamic nature of resource request.

HadiGoudarzi et al. in paper [19] proposed a RAS by categorizing the cluster in the system based on the number and type of computing, data storage and communication resources that they control. All of these resources are allocated within each server. The disk resource is allocated based on the constant need of the clients and other kind of resources in the servers and clusters are allocated using Generalized Processor Sharing (GPS). This system performs distributed decision making to reduce the decision time by parallelizing the solution and used greedy algorithm to find the best initial solution. The solution could be improved by changing resource allocation. But this system cannot handle large changes in the parameters which are used for finding the solution.

3.7 Auction

Cloud resource allocation by auction mechanism is addressed by Wei-Yu Lin et al. in [24]. The proposed mechanism is based on sealed-bid auction. The cloud service provider collects all the users' bids and determines the price. The resource is distributed to the first kth highest bidders under the price of the (k+1)th highest bid. This system simplifies the cloud service provider decision rule and the clear cut allocation rule by reducing the resource problem into ordering problem. But this mechanism does not ensure profit maximization due to its truth telling property under constraints.

The aim of resource allocation strategy is to maximize the profits of both the customer agent and the resource agent in a large datacenter by balancing the demand and supply in the market. It is achieved by using market based resource allocation strategy in which equilibrium theory is introduced (RSA-M) [25]. RSA-M determines the number of fractions used by one VM and can be adjusted dynamically according to the varied resource requirement of the workloads. One type of resource is delegated to publish the resource's price by resource agent and the

resource delegated by the customer agent participates in the market system to obtain the maximum benefit for the consumer. Market Economy Mechanism is responsible for balancing the resource supply and demand in the market system.

3.8. Application

Resource Allocation strategies are proposed based on the nature of the applications in [26]. In the work by Tram et al. [26], Virtual infrastructure allocation strategies are designed for workflow based applications where resources are allocated based on the workflow representation of the application. For work flow based applications, the application logic can be interpreted and exploited to produce an execution schedule estimate. This helps the user to estimate the exact amount of resources that will be consumed for each run of the application. Four strategies such as Naive, FIFO, Optimized and services group optimization are designed to allocate resources and schedule computing tasks.

Real time application which collects and analyzes real time data from external service or applications has a deadline for completing the task. This kind of application has a light weight web interface and resource intensive back end. To enable dynamic allocation of cloud resources for back-end mashups, a prototype system is implemented and evaluated for both static and adaptive allocation with a test bed cloud to allocate resources to the application. The system also accommodates new requests despite a-priori undefined resource utilization requirements. This prototype works by monitoring the CPU usage of each virtual machine and adaptively invoking additional virtual machines as required by the system.

David Irwin et al. [27] have suggested the integration of high bandwidth radar sensor networks with computational and storage resources in the cloud to design end-to-end data intensive cloud systems. Their work provides a platform that supports a research on broad range of heterogeneous resources and overcomes the challenges of coordinated provisioning between sensors networks, network providers and cloud computing providers. Inclusion of non-traditional resources like Steerable sensors and cameras and stitching mechanisms to bind the resources are the requirement of this project.

V. ADVANTAGES AND LIMITATIONS

There are many benefits in resource allocation while using cloud computing irrespective of size of the organization and business markets. But there are some limitations as well, since it is an evolving technology. Let's have a comparative look at the advantages and limitations of resource allocation in cloud.

4.1. Advantages:

- ✓ The biggest benefit of resource allocation is that user neither has to install software nor hardware to access the applications, to develop the application and to host the application over the internet.
- ✓ The next major benefit is that there is no limitation of place and medium. We can reach our applications and data anywhere in the world, on any system.
- ✓ The user does not need to expend on hardware and software systems.
- ✓ Cloud providers can share their resources over the internet during resource scarcity.

4.2 Limitations

- ✓ Since users rent resources from remote servers for their purpose, they don't have control over their resources.
- ✓ Migration problem occurs, when the users wants to switch to some other provider for the better storage of their data. It's not easy to transfer huge data from one provider to the other.
- ✓ In public cloud, the clients' data can be susceptible to hacking or phishing attacks. Since the servers on cloud are interconnected, it is easy for malware to spread.
- ✓ Peripheral devices like printers or scanners might not work with cloud. Many of them require software to be installed locally. Networked peripherals have lesser problems.
- ✓ More and deeper knowledge is required for allocating and managing resources in cloud, since all knowledge about the working of the cloud mainly depends upon the cloud service provider.

VI. CONCLUSION

Cloud computing technology is increasingly being used in enterprises and business markets. In cloud paradigm, an effective resource allocation strategy is required for achieving user satisfaction and maximizing the profit for cloud service providers. This paper summarizes the classification of RAS and its impacts in cloud system. Some of the strategies discussed above mainly focus on CPU, memory resources but are lacking in some factors. With this paper, we make a noteworthy commitment towards building an asset administration middleware for cloud conditions. We recognize a key segment of such a middleware and present a convention that can be utilized to meet our outline objectives for asset administration: decency of asset distribution concerning destinations, effective adjustment to load changes and versatility of the middleware layer as far as both the quantity of machines in the cloud and additionally the quantity of facilitated locales/applications. We exhibited a babble convention P* that registers, in a dispersed and consistent design, a heuristic answer for the asset portion issue for a powerfully changing asset request.

REFERENCES

[1] M. Armbrust et al., "Above the Clouds: A Berkeley View of Cloud Computing," technical report, Univ. of California, Berkeley, Feb. 2009.
[2] L. Siegele, "Let It Rise: A Special Report on Corporate IT," The Economist, vol. 389, pp. 3-16, Oct. 2008.
[3] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield, "Xen and the Art of Virtualization," Proc. ACM Symp. Operating Systems Principles (SOSP '03), Oct. 2003.
[4] "Amazon elastic compute cloud (Amazon EC2)," http://aws. amazon.com/ec2/, 2012.
[5] C. Clark, K. Fraser, S. Hand, J.G. Hansen, E. Jul, C. Limpach, I. Pratt, and A. Warfield, "Live Migration of Virtual Machines," Proc. Symp. Networked Systems
Design and Implementation (NSDI '05), May 2005.
[6] M. Nelson, B.-H. Lim, and G. Hutchins, "Fast Transparent Migration for Virtual Machines," Proc. USENIX Ann. Technical Conf., 2005.
[7] M. McNett, D. Gupta, A. Vahdat, and G.M. Voelker, "Usher: An Extensible Framework for Managing Clusters of Virtual Machines," Proc. Large Installation System Administration Conf. (LISA '07), Nov. 2007.
[8] T. Wood, P. Shenoy, A. Venkataramani, and M.Yousif, "Black-Box and Gray-Box Strategies for Virtual Machine Migration," Proc. Symp. Networked Systems
Design and Implementation (NSDI '07), Apr. 2007.
[9] C.A. Waldspurger, "Memory Resource Management in VMware ESX Server," Proc. Symp. Operating SystemsDesign and Implementation (OSDI '02), Aug. 2002.
[10] Microsoft Inc., http://www.microsoft.com/windowsazure/, Feb. 2012.
[11] M. Jelasity, A. Montresor, and O. Babaoglu, "Gossip-based aggregation in large dynamic networks," *ACM Trans. Computer Syst.*, vol. 23, no. 3, pp. 219–252, 2005.
[12] "T-Man: gossip-based fast overlay topology construction," *Computer Networks*, vol. 53, no. 13, pp. 2321–2339, 2009.
[13] F. Wuhib, R. Stadler, and M. Spreitzer, "Gossip-based resource management for cloud environments," in *2010 International Conference on Network and Service Management*.
[14] F. Wuhib, M. Dam, R. Stadler, and A. Clem, "Robust monitoring of network-wide aggregates through gossiping," *IEEE Trans. Network and Service Management*, vol. 6, no. 2, pp. 95–109, June 2009.

[15] F. Wuhib, M. Dam, and R. Stadler, "A gossiping protocol for detecting global threshold crossings," *IEEE Trans. Network and Service Management*, vol. 7, no. 1, pp. 42–57, Mar. 2010.
[16] G. Pacifici, W. Segmuller, M. Spreitzer, and A. Tantawi, "Dynamic estimation of CPU demand of web traffic," in *2006 International Conference on Performance Evaluation Methodolgies and Tools*.
[17] Z. Gong, X. Gu, and J. Wilkes, "PRESS: PRedictive Elastic ReSource Scaling for cloud systems," in *2010 International Conference on Network and Service Management*.
[18] D. Carrera, M. Steinder, I. Whalley, J. Torres, and E. Ayguade, "Utilitybased placement of dynamic web applications with fairness goals," in *2008 IEEE Network Operations and Management Symposium*.
[19] S. Voulgaris, D. Gavidia, and M. van Steen, "CYCLON: inexpensive membership management for unstructured p2p overlays," *J. Network and Systems Management*, vol. 13, no. 2, pp. 197–217, 2005.

[20] R. L. Graham, "Bounds on multiprocessing timing anomalies," *SIAM J. Applied Mathematics*, vol. 17, no. 2, pp. pp. 416–429, 1969.
[21] C. Tang, M. Steinder, M. Spreitzer, and G. Pacifici, "A scalable application placement controller for enterprise data centers," in *2007 International Conference on World Wide Web*.
[22] H. Shachnai and T. Tamir, "On two class-constrained versions of the multiple knapsack problem," *Algorithmica*, vol. 29, no. 3, pp. 442–467, Dec. 2001.
[23] G. B. Dantzig, "Discrete-variable extremum problems," *Operations Research*, vol. 5, no. 2, pp. 266–288, 1957.
[24] L. Breslau, P. Cao, L. Fan, G. Phillips, and S. Shenker, "Web caching and Zipf-like distributions: evidence and implications," in *Proc. 1999 Annual Joint, 134*
[25] C. Adam and R. Stadler, "Service middleware for self-managing largescale systems," *IEEE Trans. Network and Service Management*, vol. 4, no. 3, pp. 50–64, Apr. 2008.
[26] J. Famaey, W. De Cock, T. Wauters, F. De Turck, B. Dhoedt, and P. Demeester, "A latency-aware algorithm for dynamic service placement in large-scale overlays," in *2009 International Conference on Integrated Network Management*.
[27] C. Low, "Decentralised application placement," *Future Generation Computer Systems*, vol. 21, no. 2, pp. 281–290, 2005.
[28] Y. Yazir, C. Matthews, R. Farahbod, S. Neville, A. Guitouni, S. Ganti, and Y. Coady, "Dynamic resource allocation in computing clouds using distributed multiple criteria decision analysis," in *2010 IEEE International Conference on Cloud Computing*.
[29] E. Loureiro, P. Nixon, and S. Dobson, "Decentralized utility maximization
for adaptive management of shared resource pools," in *2009 International Conference on Intelligent Networking and Collaborative Systems*.
[30] G. Cybenko, "Dynamic load balancing for distributed memory multiprocessors," *J. Parallel and Distrib. Computing*, vol. 7, no. 2, pp. 279–301,
1989.
[31] R. Elsasser, B. Monien, and R. Preis, "Diffusion schemes for load balancing on heterogeneous networks," *Theory of Computing Systems*, vol. 35, p. 2002, 2002.