Chapter 1

Introduction

1.1 Problem Statement

Cloud computing is a computing model offering a network of servers to their clients in a on-demand fashion. From NIST's definition [14], *"cloud computing is a model for enabling ubiquitous, convenient, on-demand network access to a shared pool of configurable computing resources (e.g., networks, servers, storage, applications and services) that can be rapidly provisioned and released with minimal management effort or service provider interaction."* To illustrate how it works, considering a case: a Cloud provider builds a data center which contains thousands of servers connected with network. These servers are virtualized which means they are partitioned into a smaller unit of resources called *Virtual Machines (VMs)*. A web-based application provider can access and deploy their applications (e.g Endnote, Google Drive and etc.) in these VMs from anywhere in the world. Once the applications start serving, application users can use them without installing on their local computers.

Generally, Cloud computing involves three stakeholders: Cloud providers, Cloud users, and End (application) users [9] (see Figure 1.1).

Each stakeholder has their responsibility, goal, and objectives.

- *Cloud providers* build data centers, provide maintenance and resource management on the hardware infrastructure. Their goal is to increase the profit by boosting the income and reducing the expense. Their income comes from Cloud users' rental of servers or *Physical Machines (PMs)* in terms of resource quality (e.g 3.5GHz dual-core CPU), quantity (e.g 3 PMs), and time (e.g 1 year). Therefore, Cloud providers objective is to maximize utilization of computing resources. A high utilization brings two benefits, firstly, it increases income by accommodating more applications in limited resources. Secondly, it cuts the expense of energy consumption by packing applications in a minimum number of PMs so that idle PMs can be turned off.

- *Cloud users* develop and deploy applications on Cloud. Each application generates time-vary CPU utilization. Their goal is also increase the profit mainly through two objectives, attracting more End users and reduce the expense of resources. The first objective can be achieved by improving the quality of service as well as lower the fee for End users. Either way depends not only on the software entities but also the quality of service (QoS) offered by Cloud provider. The second objective can be achieved by a good estimation of the reserved resources, so that they do not rent insufficient or too much resources which cause performance degradation or wastage.

- *End Users* are the final customers in this chain. They consume services directly from Cloud users and indirectly from Cloud provider. Their goal is to obtain a satisfactory

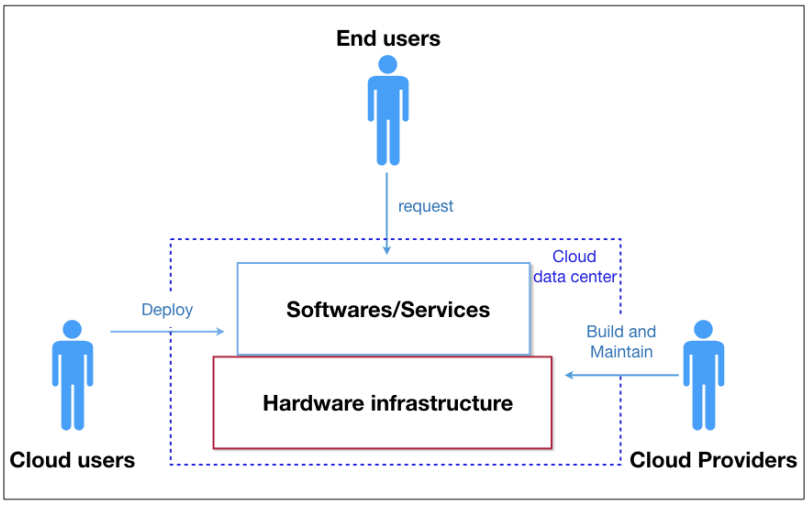

Figure 1.1: Stakeholders of Cloud computing

service. It is achieved by signing a Service Level Agreement (SLA) with Cloud users which constrains the performance of the services.

In this thesis, we focus on an core issue of helping Cloud providers to increase their profits by optimizing the resource allocation in data centers. Specifically, the profit can be improved by reducing the energy consumption. A direct way to reduce energy consumption is to always use a minimum number of Physical machines (PMs) hosting applications. This is because the more PMs are running, the more energy they are consumed.

The problem can be described as, a data center has a number of PMs where each of them can be represented as a set of resources such as CPU cores, RAM and etc. The data center received a list of requests for applications which is also represented as resources. The task is to allocate these requested resources to a minimum number of PMs. The decision variable in this task is the location of each requested resource. The assumptions and constraints are distinct in service models of Cloud computing.

Service models of Cloud computing are critical in solving energy consumption problem because their distinct ways of managing resources have sever effect on the problem. These distinct ways of resource management mainly result from the responsibilities among stakeholders. There are three traditional service models [14]: Infrastructure as a Service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS). The relationship among three service models can be pictured as a pyramid (see 1.2). Their detailed responsibilities among stakeholders are explained as below:

- IaaS, a Cloud provider hosts hardwares such as PMs and cooling infrastructure on behalf of Cloud users. A Cloud provider also provides maintenance, management of hardware resources.

  In IaaS, Cloud provider offers the fundamental resources such as CPU cores, RAMs, and network bandwidth. These resources are often encapsulated in virtualized computing units called virtual machines (VMs). Cloud providers establish a number of types of VM for simplifying the management. The 'type' means a VM contains a certain quantity of resources such as 2-cores and 1 GB RAM. *Traditional* IaaS and PaaS use VM as the fundamental unit of resources.

  A typical procedure of a Cloud user deploying their applications in an IaaS cloud includes several steps. Initially, Cloud users estimate the resources that their appli-

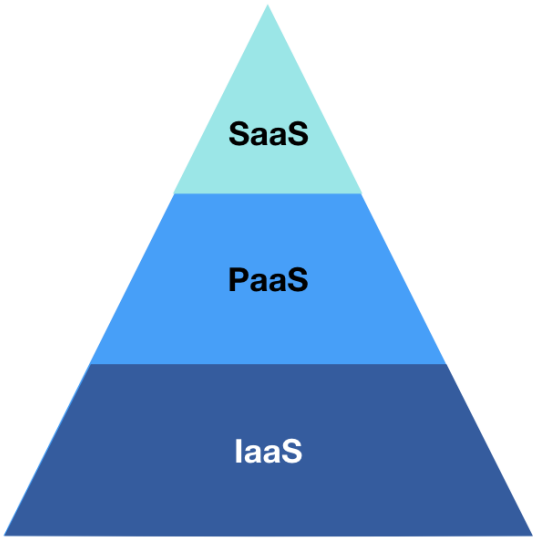

Figure 1.2: Pyramid of service models in Cloud computing. IaaS provides the fundamental resources such as CPU cores, RAM. The resources are usually wrapped with various types of virtual machine. PaaS provides a software platform sitting on top of IaaS for Cloud users to develop applications. SaaS is the application that serves End Users.

cations might consume and select a type of VM which can satisfy the requirement. After Cloud users have made the decisions, they send requests to Cloud providers for a number of VMs. Finally, Cloud providers received the request, provisioned and allocated these VMs to PMs.

From the resource management perspective, the constraint in allocation of VMs is that the aggregated VMs' resources in a PM cannot exceed the capacity of the PM. After these VMs have been allocated, their types cannot be changed. During the life cycle of an application, Cloud providers can dynamically adjust the locations of VMs, provision new VMs (same type) for the replicas of an application, as well as turning on/off PMs.

- PaaS, a Cloud provider offers a platform which allows Cloud users to develop, test and deploy their applications on it.

  From resource management perspective, as shown in Figure 1.2, PaaS is sitting above the IaaS which means the underlying resource is still based on IaaS VM types. Different from IaaS, PaaS takes the responsibility of selecting VMs and allows Cloud users to focus on software development.

  A typical procedure of a Cloud user deploying their applications in an PaaS cloud includes several steps. In the first step, Cloud users need to provide the initial estimation of the quantity of resources instead of types of VM. Then, Cloud providers determine the types of VM for applications according to the estimated resources. After this step, resource management system conducts the provisioning and allocating as the same steps in IaaS. During the life cycle of applications, similar to IaaS, Cloud providers can also adjust the location of VMs, add new VMs, and control the status of PMs. Different from IaaS, Cloud providers can change the type of VM for an application as long as the application's performance can be guaranteed.

- SaaS describes the relationship between Cloud users and End users. End users create

workloads for applications. Although this service model does not directly related to the resource management, it provides the fundamental reasons for resource management and optimization. Because of the dynamic nature of workloads, the underlying resources must also be dynamic adjusted to meet the requirement.

Our proposed optimization approaches are based on a new service model: Container as a Service (CaaS) [18] which is a variant of PaaS. The reasons for us to establish our approaches on CaaS are listed as followed:

- CaaS uses a new virtualization technology called containers which has shown several important characteristics that overcome the disadvantages of traditional IaaS and PaaS, therefore, CaaS is a promising trend.

- CaaS has a fine granularity level of resource management which has shown opportunities to improve the resource utilization, however, it often proposes new optimization challenges which have not been studied yet.

Firstly, we illustrate the disadvantages of traditional IaaS and PaaS and discuss the reasons of why current approaches cannot completely overcome these problems. From there, we explain why CaaS is a prescription of their problems.

IaaS has three characteristics which naturally lead to a low resource utilization.

- Separated responsibilities of resource selection for applications and resource allocation. As above mentioned, Cloud users have to select the type of resources. This causes a problem. The accurate estimation is almost impossible because of unpredictable workloads; Cloud users tend to reserve more resources for ensuring the QoS at the peak hours [5] than completely rely on auto-scaling, simply because auto-scaling is more expensive than reservation. However, the peak hours only account for a short period, therefore, in most of time, resources are wasted. As the types of VM are a part of the contract, Cloud providers cannot simply change the type of VMs after provisioning.

- Cloud providers offer fixed types of VM. Because of the fixed size of resources and the one-on-one mapping of applications and VMs, specific applications consume unbalanced resources which leads to vast amount of resource wastage [24]. For example, computation intensive tasks consume much more CPU than RAM; a fixed type of VM provides much more RAM than it needs. Because the tasks use too much CPU, they prevent other tasks from co-allocating. This also causes wastage.

- Redundant operating systems (OSs) cause vast resource wastage. Since each VM runs a separated operating system, a PM might run many operating systems at a time. However, normal applications do not need specific operating systems commonly used OSs - such as Linux-based: RedHat, or Windows server versions - are well enough for their needs. Therefore, running duplicated OSs in a PM is unnecessary.

For the first two drawbacks, previous researchers and developers have come up with two strategies: server consolidation and overbooking.

Server consolidation [29], as above mentioned, utilized a dynamic migration technique to resolve the low utilization problem by gathering applications into a fewer number of PMs (see Figure **??**), so that the resource utilization of PMs are maintained at a high level. In the meanwhile, idle PMs can be turned off to save energy. Consolidation dramatically improves hardware utilization and lowers PM and cooling energy consumption.

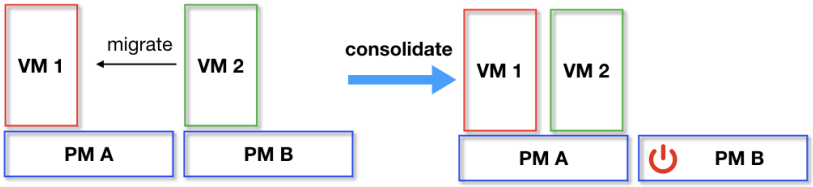

Figure 1.3: A Server Consolidation example: Initially, each PM runs an application wrapped with a VM in a low resource utilization state. After the consolidation, both VMs are running on PM A, so that PM B is turned off to save energy [2].

Overbooking strategy [24] is used to overcome the low utilization problem raised by the over-provisioning of VMs. It allocates more VMs in a PM even their aggregated resources have exceeded the PM's capacity. The advantage of this approach is that, indeed, it improves the resource utilization. The disadvantages are also obvious: if one or more applications are experiencing a heavy load; the PM is likely to run out of resources. At this moment, all the applications are suffer from a QoS degradation. In order to avoid the overloading, Cloud users often carefully predict the utilization of applications based on their previous workloads, when applications start degrading, a dynamic resource management approach is used to adjust the VMs' location.

The workflow of utilizing these two strategies in resource management is shown in Figure **??**. It seemingly solves the over-provisioning problem, however, the overbooking strategy brings new challenges. Although the overbooking strategy have been studies for years [], the prediction of workload is very difficult [] or even impossible as referred in many literatures []. The improvement of resource utilization is completely based on an accurate prediction of a large number of applications. Its effectiveness is hard to evaluate and justify. It is also very time consuming to predict the utilization of every application. Therefore, it is urgent to provide a strategy without making a prediction. One possible way is to use finer granularity strategy of resource management.

Furthermore, there is no solution for the third drawback in the context of IaaS.

For traditional PaaS, Cloud providers can adjust the applications' location and the type of VM. Therefore, it overcomes the first drawback of IaaS. Because of PaaS is built upon IaaS, the one-on-one relationship between application and VM still exist. PaaS can only select the most suitable type instead of changing their sizes. Therefore, the unbalanced resource problem cannot be solved. In addition, PaaS brings a restriction for the applications deployed on it. PaaS build a software middle-ware to allow Cloud users' development. The middle-ware requires the deployed applications to be compatible with the environment, for example, Google App engine [] only allows certain programming languages and libraries. Therefore, the generality of PaaS is limited. It is urgent to provide a environment which supports automatic resource management as well as an editable programming environment.

The recent development of container technology [21] has driven the attention of both industrial and academia because its potential to solve these problems in both IaaS and PaaS. Container is an operating system level of virtualization which means multiple containers can be installed in a same operating system (see Figure 1.4 right-hand side). Each container provides an isolated environment for an application. In short, a VM is partitioned into smaller manageable units. Container as a Service (CaaS) is a variant of PaaS propose by a leading Lab in this field []. Instead of VMs, CaaS uses both container and VM as a hybrid resource management unit.

CaaS solves the problems in IaaS and PaaS because of containers' characteristics. The most important feature of container is that it utilizes the resources inside VMs. Therefore,

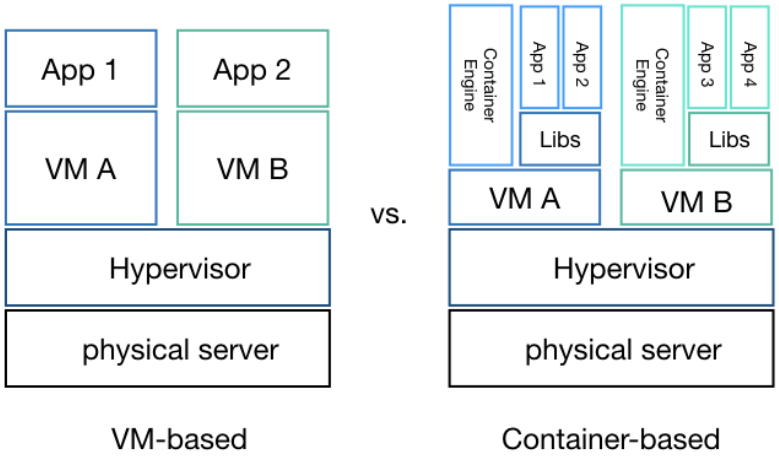

Figure 1.4: A comparison between VM-based and Container-based virtualization

even CaaS is also built upon IaaS, it solves the fixed types by allocating containers in it. This feature also solves the redundant operating systems problem. The second characteristic is that a container's size is changeable and can be controlled by Cloud provider in real-time. Therefore, the Cloud providers can monitor the resource utilization of applications and adjust it accordingly. This feature provides better flexibility which solves the over-provisioning problem. In addition, one feature that separates CaaS and PaaS is that, containers allow user-defined libraries instead of constrained by the platform.

Although CaaS has these promising characteristics, it also proposes a difficult research problem. In order to understand the difficulty of the problem, it is necessary to illustrate the difficulty of current VM-based server consolidation problem.

Currently, vast amount of server consolidation methods are mostly VM-based and they are mainly modeled as bin-packing problems [13]. This is because VMs and PMs are naturally modeled as items and bins and server consolidation and bin-packing have the same optimization objective: minimize the number of bins/PMs. The complexity of bin-packing problem is NP-hard which means it is extreme time-consuming to find its optimal solution when the number of decision variables are large. Deterministic methods such as Integer Linear Programming [23] and Mixed Integer Programming [25] have been used but they are well-known that they are very time-consuming for a large scale problem. More research proposed heuristic methods to approximate the optimal solution such as First Fit Decreasing (FFD) [17], Best Fit Decreasing (BFD) [3]. Manually designed heuristics are designed to tackle the special requirements such as a bin-item incomplete scenario [7] and multi-tier applications [11, 12]. Although these greedy-based heuristics can quickly approximate an answer, as Mann's research [13] showed, server consolidation is a lot more harder than bin-packing problem because of the multi-dimensional, many constraints. Therefore, general bin-packing algorithms do not perform well with many constraints and specific designed heuristics only perform well in very narrow scope.

Although traditional VM-based server consolidation has been studied for year, these methods can not be directly applied on container-based problem because container-based consolidation has two levels of allocation: Containers are allocated to VM as the first level and VMs are allocated to PMs as the second level. These two levels of allocation interact with each other, for the first level of allocation affects the decision in the second level.

1.2 Motivation

Cloud data center has a high dynamic nature where it constrantly receives new requests for resource provisioning and releases old current resources. Therefore, a data center needs different strategies to handle different scenarios.

In this thesis, we aims at providing a series of approaches to continuously optimize the a mixed allocation of VMs and containers. A continuous optimization procedure with three stages: initialization, global consolidation, and dynamic consolidation. Different stages have distinctive goals, therefore, they are considered as separated research questions. In addition, a scalability problem of static optimization is considered as an optional objective.

1. The initialization,
   at this stage, a set of applications or containers is allocated to a set of empty VMs and these VMs are allocated to a set of PMs. This two-step procedure is interconnected meaning the results of first level of allocation affects the second level of allocation. Therefore, both allocations should be conducted simultaneously. This problem is inherently more difficult than previous VM-based consolidation problem. VM-based consolidation is modeled as bin-packing which is NP-hard. In contrast, container-based consolidation has two levels of bin-packing, this is derived from the problem's nature. Most importantly, these two levels of problem interact and therefore can not be solved separately. This is the first research that consider server consolidation has a bi-level optimization problem [26].

   This stage will establish the fundamental concepts in studying the joint allocation of containers and VMs including new problem models: price and power model, new problem constraints, and optimization objectives. The major challenges for this objective is to design representations and several EC approaches to solve this problem. More specifically, in designing the EC approach, new search mechanisms, operators will be designed and new representations will be proposed to fit the problem.

2. A *Global consolidation* is conducted to improve the global energy efficiency at a certain time point, e.g. a fixed time interval. The challenges are three folds, firstly, similar with initialization problem, the problem has two level of allocations and they interact with each other. It is more complex than a single-level VM-based consolidation. Secondly, like VM-based consolidation, Container-based consolidation is considered as a multi-objective problem with minimization of migration cost as well as keeping a good energy efficiency. Thirdly, consolidation is a time-dependent process which means the previous solution affects the next one. Previous research only consider each consolidation as an independent process. As a consequence, although in current consolidation, the migration is minimized. It may lead to more migration in the future. We will consider the robustness of consolidation and propose a novel time-series-aware server consolidation which takes the previous consolidations and the future consolidation into consideration.

3. Dynamic consolidation continuously maintains the data center to a high energy efficiency. It is applied on single container at any time point. As mentioned in previous Chapter, dynamic placement is directly related to consolidation. Therefore, we focus on this question. To solve a dynamic placement with large number of variables, heuristics and dispatching rules are often used [19, 20, 6, 3]. In this scenario, a dispatching rule is considered as a function that determines the priorities of PMs that a container can be placed. However, dynamic placement is much complex than bin-packing problem [13]. Because of its dynamic nature, human designed heuristics are ill-equipped

in approximating solutions when the environment has changed [22]. Multi-objective genetic algorithm (GA) [27] has been applied. However, GA is too slow for dynamic problem.

We intend to develop a hyper-heuristic method - Genetic Programming (GP) technique [1] or artificial immune system [8]- to learn from the best previous allocation and automatic evolves dispatching rules to solve this problem. GP has been applied in generating dispatching rules for bin-packing problem [4, 22] and other scheduling problems [16]. The results have shown promising results.

There are mainly two challenges, first, it is difficult to identify the related factors that construct the heuristic. Factors or features are the building blocks of heuristics. It is a difficult task because the relationship between a good heuristic and features are not obvious. Second, representations provide different patterns to construct dispatching rules. It is also unclear what representation is the most suitable for the consolidation problem.

4. Cloud data center typically has hundreds of thousands PMs and more. Large scale of static server consolidation has always been a challenge since it takes large amount of variables into consider. Many approaches have been proposed in the literature to resolve the problem. There are mainly two ways, both rely on distributed methods, hierarchical-based [10, 15] and agent-based management systems [28]. The major problem in agent-based systems is that agents rely on heavy communication to maintain a high-level utilization. Therefore, it causes heavy load in the networking. Hierarchical-based approaches are the predominate methods. In essence, these approaches are centralized methods where all the states of machines within its region are collected and analyzed. The major disadvantage of hierarchical-based approaches is that it only provides local solutions. In fact, it is infeasible and unnecessary to check all the states of machines since the search space is too large and most machines do not need a change. This idea motivates a way to improving the effectiveness is to reduce the number of variables so that the search space is narrowed. In this thesis, we are going to investigate the way to eliminate the redundant information.

Bibliography

[1] BANZHAF, W., NORDIN, P., KELLER, R. E., AND FRANCONE, F. D. Genetic programming: an introduction, 1998.

[2] BARROSO, L. A., AND HÖLZLE, U. The Case for Energy-Proportional Computing. *IEEE Computer 40*, 12 (2007), 33–37.

[3] BELOGLAZOV, A., ABAWAJY, J. H., AND BUYYA, R. Energy-aware resource allocation heuristics for efficient management of data centers for Cloud computing. *Future Generation Comp. Syst. 28*, 5 (2012), 755–768.

[4] BURKE, E. K., HYDE, M. R., AND KENDALL, G. Evolving Bin Packing Heuristics with Genetic Programming. *PPSN 4193*, Chapter 87 (2006), 860–869.

[5] CHAISIRI, S., LEE, B.-S., AND NIYATO, D. Optimization of Resource Provisioning Cost in Cloud Computing. *IEEE Trans. Services Computing 5*, 2 (2012), 164–177.

[6] FORSMAN, M., GLAD, A., LUNDBERG, L., AND ILIE, D. Algorithms for automated live migration of virtual machines. *Journal of Systems and Software 101* (2015), 110–126.

[7] GUPTA, R., BOSE, S. K., SUNDARRAJAN, S., CHEBIYAM, M., AND CHAKRABARTI, A. A Two Stage Heuristic Algorithm for Solving the Server Consolidation Problem with Item-Item and Bin-Item Incompatibility Constraints. *IEEE SCC* (2008).

[8] HOFMEYR, S. A., AND FORREST, S. Architecture for an Artificial Immune System. *Evolutionary Computation 8*, 4 (2000), 443–473.

[9] JENNINGS, B., AND STADLER, R. Resource Management in Clouds: Survey and Research Challenges. *Journal of Network and Systems Management 23*, 3 (2015), 567–619.

[10] JUNG, G., HILTUNEN, M. A., JOSHI, K. R., SCHLICHTING, R. D., AND PU, C. Mistral - Dynamically Managing Power, Performance, and Adaptation Cost in Cloud Infrastructures. *ICDCS* (2010), 62–73.

[11] JUNG, G., JOSHI, K. R., HILTUNEN, M. A., SCHLICHTING, R. D., AND PU, C. Generating Adaptation Policies for Multi-tier Applications in Consolidated Server Environments. *ICAC* (2008).

[12] LI, B., AND JIANXIN, L. *An Energy-saving Application Live Placement Approach for Cloud Computing Environment*. ACM International Conference on Cloud Computing, 2009.

[13] MANN, Z. Á. Approximability of virtual machine allocation: much harder than bin packing.

[14] MELL, P. M., AND GRANCE, T. The NIST definition of cloud computing. Tech. rep., National Institute of Standards and Technology, Gaithersburg, MD, Gaithersburg, MD, 2011.

[15] MOENS, H., FAMAEY, J., LATRÉ, S., DHOEDT, B., AND DE TURCK, F. Design and evaluation of a hierarchical application placement algorithm in large scale clouds. *Integrated Network Management* (2011), 137–144.

[16] NGUYEN, S., ZHANG, M., JOHNSTON, M., AND TAN, K. C. Automatic Design of Scheduling Policies for Dynamic Multi-objective Job Shop Scheduling via Cooperative Coevolution Genetic Programming. *IEEE Transactions on Evolutionary Computation 18*, 2 (2014), 193–208.

[17] PANIGRAHY, R., TALWAR, K., UYEDA, L., AND WIEDER, U. Heuristics for vector bin packing. *research microsoft com* (2011).

[18] PIRAGHAJ, S. F., CALHEIROS, R. N., CHAN, J., DASTJERDI, A. V., AND BUYYA, R. Virtual Machine Customization and Task Mapping Architecture for Efficient Allocation of Cloud Data Center Resources. *Comput. J. 59*, 2 (2016), 208–224.

[19] SARIN, S. C., VARADARAJAN, A., AND WANG, L. A survey of dispatching rules for operational control in wafer fabrication. *Production Planning and ... 22*, 1 (Jan. 2011), 4–24.

[20] SHI, W., AND HONG, B. Towards Profitable Virtual Machine Placement in the Data Center. *UCC* (2011), 138–145.

[21] SOLTESZ, S., PÖTZL, H., FIUCZYNSKI, M. E., BAVIER, A. C., AND PETERSON, L. L. Container-based operating system virtualization - a scalable, high-performance alternative to hypervisors. *EuroSys 41*, 3 (2007), 275–287.

[22] SOTELO-FIGUEROA, M. A., SOBERANES, H. J. P., CARPIO, J. M., HUACUJA, H. J. F., REYES, L. C., AND SORIA-ALCARAZ, J. A. Evolving Bin Packing Heuristic Using Micro-Differential Evolution with Indirect Representation. *Recent Advances on Hybrid Intelligent Systems 451*, Chapter 28 (2013), 349–359.

[23] SPEITKAMP, B., AND BICHLER, M. A mathematical programming approach for server consolidation problems in virtualized data centers. *IEEE Transactions on services ...* (2010).

[24] TOMÁS, L., AND TORDSSON, J. Improving cloud infrastructure utilization through overbooking. *CAC* (2013), 1.

[25] WANG, Y., AND XIA, Y. Energy Optimal VM Placement in the Cloud. In *2016 IEEE 9th International Conference on Cloud Computing (CLOUD* (2016), IEEE, pp. 84–91.

[26] WEN, U.-P., AND HSU, S.-T. Linear Bi-Level Programming Problems – A Review. *The Journal of the Operational Research Society 42*, 2 (Feb. 1991), 125.

[27] XU, J., AND FORTES, J. A. B. Multi-Objective Virtual Machine Placement in Virtualized Data Center Environments. *GreenCom/CPSCom* (2010).

[28] YAZIR, Y. O., MATTHEWS, C., FARAHBOD, R., NEVILLE, S. W., GUITOUNI, A., GANTI, S., AND COADY, Y. Dynamic Resource Allocation in Computing Clouds Using Distributed Multiple Criteria Decision Analysis. *IEEE CLOUD* (2010), 91–98.

[29] ZHANG, Q., CHENG, L., AND BOUTABA, R. Cloud computing: state-of-the-art and research challenges. *Journal of Internet Services and Applications 1*, 1 (2010), 7–18.