A General and Practical Consolidation Framework in CloudNFV

Chuan Pham*, Hoang Dai Tran†, Seung Il Moon‡, Kyi Thar§, and Choong Seon Hong¶
Department of Computer Engineering
Kyung Hee University
1 Seocheon, Giheung, Yongin, Gyeonggi, 449-701 Korea
Email: {*pchuan, †dai.tran, ‡moons85, §kyithar, ¶cshong}@khu.ac.kr

***Abstract*—In cloud computing environment, power consumption is a critical factor for data centers. Effective power consumption control can improve significantly benefit for providers. In this paper, we derive from CloudNFV architecture, an open platform for implementing Network Functions Virtualization based on cloud computing and Software Defined Networking, to obtain an optimal power control strategy toward reducing CO_2 emissions and maximize total utility of active resources. We propose a consolidation function for determining the On/Off strategy of each server and use live migration method to rearrange virtual machines in active resource components. The efficiency of our proposed model is demonstrated on the simulation environment.**

I. Introduction

CloudNFV is a platform to test the integration of cloud computing, software defined networking (SDN), network functions virtualization (NFV), and the Telemanagement Forum (TMF) management in an open, real-world way [1], [2]. This is an open platform for implementing NFV based on cloud computing and software defined network (SDN) technogolies in a multi-vendor environment. Through this combination, operators can now simplify their networks, remove the complexities of topology and service creation, and accelerate the process of new service creation and delivery. The main idea in CloudNFV is to virtualize everything, from resources to services to functions, even to the users themselves. In the classical network appliance approach, a considerable amount of resources is waste from fragmented non-commodity hardware, physical install per appliance per site, etc. From the point of view "everything is virtual", CloudNFV aims to reduce costs or at raising revenues, maximize operational efficiencies and introduce new revenue-generating services on a scale never before possible [3].

In the previous works, many proposals focus on the optimal resource management in Cloud computing. However, in CloudNFV architecture, there is a few proposals that focus on the optimal resource management. Especially, to improve the resource utility and reduce the power consumption and stabilize whole system, server consolidation is a potential area research in cloud computing. In distributed cloud network, OpenStack is the first one, who implemented the dynamic consolidation framework in OpenStack Cloud model. By using the live migration technique, the major objective of dynamic server consolidation is to improve the utilization of physical resource and reduce energy consumption by re-allocating virtual machines (VMs) using live migration according to their real-time resource demand and switching idle hosts to the sleep mode.

The consolidation idea is proposed in [4] when they monitored the average utilization in data center. There are many servers that consume huge energy without performing useful works. Hence, they proposed a strategy that provides a solution to not only control the quality of services but also cut down the consumption of energy by data centers. The term "consolidation server" was born with this concept. Many papers in cloud computing refer from this and consider consolidation server as a problem in VM placement. Then, they try to solve by the some algorithms such as bin packing algorithm (e.g. first fit, best fit, etc.), ant colony algorithm, novel vector based approach, binary search tree, etc. All proposed algorithms always try to reduce number of required servers. However, these algorithms are focused on the general cloud computing. There is no work proposed scheme that integrates the consolidation function for CloudNFV architecture.

In this paper, we focus on how to optimize resource allocations in Network Function Virtualization Orchestration (NFVO). The CloudNFV architecture is based on management and orchestration applications built around agile data/process model called Active Virtualization. However, current management solution of CloudNFV is still an open research area that optimization theory and machine learning can be applied. In consolidation field, CloudNFV is a potential research trend. The major objective of dynamic VMs consolidation is to improve the utilization of physical resources and reduce energy consumption by re-allocation virtual machines (VMs) according to real-time resource demands and switching idle hosts to the sleep mode [5]. This function need to improve and adapt with CloudNFV architecture. For example, in CloudNFV, NFVO has enough information to analyze resource status such as: VM status, remaining resource pools , physical server utility, etc. They are significant data that make the consolidation function robust and efficient. Our work proposes a new consolidation function, which includes in the resource management optimizer component in CloudNFV.

In depth study of virtualization in cloud computing, consolidation system can reduce the power consumption and

stabilize whole system. By using live migration process in consolidation, administrators can reorganize some VMs from imbalance of resource hosts or migrate VMs to release the inefficient servers but the system still keeps VMs running without disrupting the services. However, the copy process requires more time, resources on the source and destination servers (such as: CPU, memory, storage, bandwidth, etc.) to perform. In some special conditions, live migration causes a dramatic reduction in the amount of available resources, which can easily lead to rejection of new user requests due to lack of available physical slots [6]. Therefore, choosing the best threshold for consolidating system plays a significant role that makes the whole system stable. In our work, after collecting information from applications and resource status, live migration is considered with many cases to rearrange the VMs into the best place in which the providers can archive the highest utility of servers and guarantee the consistency of all servers.

To consolidate server, live migration function is applied in moving VMs from a server to others for providing additional resources or releasing cold spot (i.e. an underloaded host) or hot spot (i.e. an overloaded host) [7], [8], [4], [9]. Consolidation supports unneeded servers to put into a low-power state or switched off or devoted to the execution of incremental workload [8]. Traditionally, bin packing problem is a suitable approach to solve the VM placement problems such as: First fit algorithm, Best fit algorithm, Exact fit algorithm, etc. However, in the bin packing problem, according to the computational complexity theory, it is a combinational NP hard problem. Especially, with many dimensions, many knapsacks with different size, this problem will be more severe. That reason causes that in many previous papers, the authors tried to transform many dimension problems into one dimension problems. Inspired from the development of [10], [11] about solving the placement problem by genetic algorithm (GA), we analyze and propose the policy of dynamic profit in data centers which is applied GA for optimizing the cost of serving.

The rest of the paper is organized as follows. The system model is represented in Section II. In Section III, we formulate our scenario and relate to the knapsack problem. The dynamic resource allocation in NFVO is represented in Section IV. In Section V, we describe the experiment environment and simulation results. Finally, we conclude our work in Section VI.

II. System model

We consider a consolidation function in NFVO [1], [2]. In [12], consolidation function does not have enough information about task arrival rate, and resource status. Therefore, this function is often implemented in physical layer that just focuses on resource management and power consumption. However, in CloudNFV environment, NFVO connects to VNF Managers component that can collect information from VMs and resource status. From these worth data, we propose four components in Resource Management Optimizer (RMO) such as: Resource Information Collection module, Analysis and Prediction Module, Optimization module and Determination module.

In the following section, we discuss the design and interaction of these components, as well as the specification process of them.

A. Components

Our work focuses on four components:

- Resource Information Collection module - a module that is deployed to collection information of resource usage in history.
- Analysis and Prediction module - a module that is deployed to predict resource utility of physical servers.
- Optimization module - a module that is deployed to compute an optimization allocation scheme.
- Determination module - module that is deployed to make management decisions.

B. Consolidation model in NFVO

Consolidating system is a periodic task, i.e. after *t* time slot (e.g. fifteen minutes or an hour), RMO checks the resource status and determines whether consolidate servers or not. Furthermore, as shown in Figure 1, when a new VM is requested by users, RMO computes a best place to host VM that can reduce consolidation cost in the future. And, when a VM need to extend more resources, RMO also determines the scheme to allocate. Our method focuses on the centralize controller, which receives all requests and resources status in physical hardware. This controller determines destination hosts to create, migrate VMs, and also is responsible for turning ON/OFF servers when necessary. The Optimization module is the main module in our model that focuses on optimizing VM placement. To solve the VM placement algorithm, we derive from the research in [10], [11]. They proposed an improvement in GA to obtain the optimal solution in multi dimension multi knapsack problem (MMKP). So, we name this function is GSC that combines GA and the technique server consolidation.

III. Problem formulation

In our scenario, we take into account the resource allocation constraints in cloud computing. We assume that in data centers, there are *N* servers and *M* VMs, which need to be rearranged to consolidate the system at this current time. Each server has k types of resource (e.g. CPU, memory, storage, communication bandwidth, etc.). The constraints of total demanding resources in VMs, which are occupying, cannot be exceeded the threshold of the containing server.

$$\sum_{j=1}^{M} R_{jk} X_{ij} \leq T_{ik} \tag{1}$$

where

$$X_{ij} = \begin{cases} 1 & \text{if } VM_j \text{ is assigned to server } i, \\ 0 & \text{otherwise,} \end{cases} \tag{2}$$

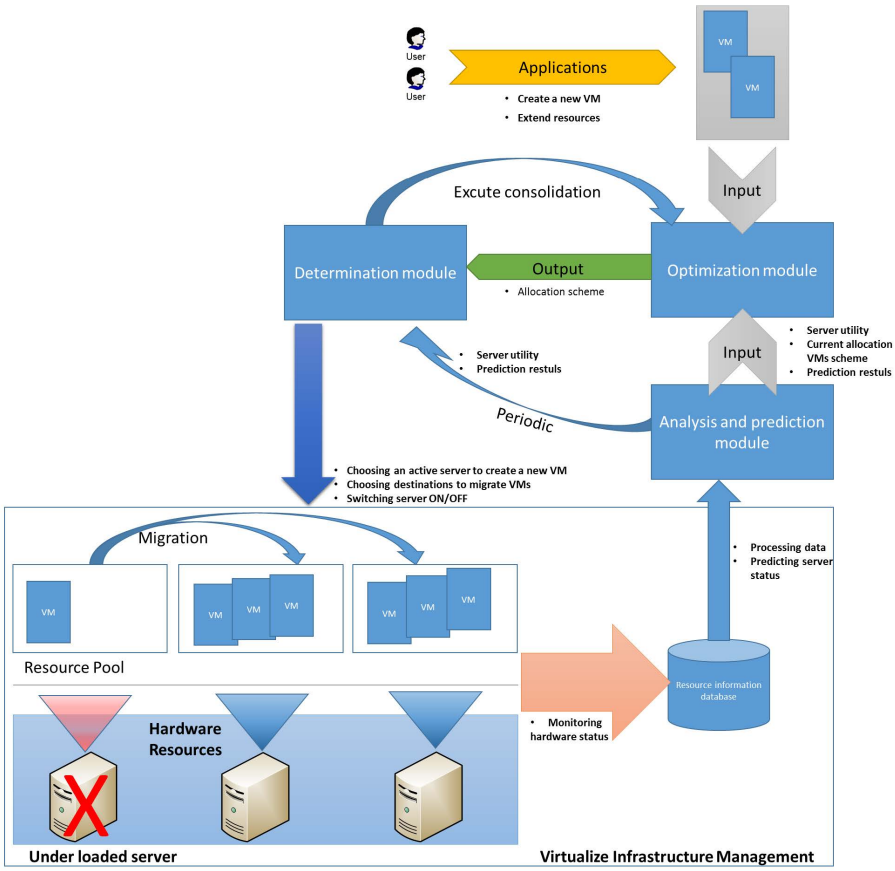

Fig. 1: Consolidation model in NFVO.

and T_{ik} is the threshold of the resource type k in server i, R_{jk} is the capacity of the resource type k of VM_j, and X is the binary allocation matrix VMs.

Secondly, some traditional mechanisms [4], [12] rearrange VMs to reduce number of active servers by considering the capacity constraints of CPU, memory and storage in VMs and servers. In the provider point of view, the revenue when hosting VMs in each server is different from others. Therefore, we consider the scenario with the dynamic serving cost of each VM in data centers.

In each time slot, the cost of each VM will be changed depend on the percentage resource load in each server. The higher resource load, the cheaper the serving cost for hosting a VM is. We define a dynamic cost for each VM which is specified particularly in the hosting server. The profit of each VM can be calculated by:

$$p_j = pr^j_{user} - pr^j_{service}, \tag{3}$$

where p_j is profit of VM_j, pr^j_{user} is the payment of users for VM_j and $pr^j_{service}$ is the cost of services for VM_j .

To calculate the cost of services $pr^j_{service}$, we consider in a discrete time slot scenario. Every time, when having a request for migration or creating a new VM, each server will be calculated the price of each VM instance. We define the price of service for each VM, a new character of VM that represents the relationship between a VM and the average utility of server. The formula of dynamic price is showed as following:

$$pr^j_{service}(t) = pr^j_{base} + (1 - \overline{r}_i(t))pr^j_{base}, \tag{4}$$

At instant time t, $pr^j(t)$ is the price of VM_j, pr^j_{base} is the basic instance price of VM_j (i.e. a fix value which can adjust by administrator) and $\overline{r}_i(t)$ is the average utilization of all resources of server i. Without loss of generally, we assume that each VM has a basic price which depends on the instance of a VM. The achieving profit when hosting VM is inversely proportional to the price of VM.

From the observation above, we represent our problems as follows: given a set of M servers and a set of M VMs with: p_{ij} is profit of VM_j in server i, R_{jk} is the capacity of the resource type k of VM_j, and T_{ik} is threshold of the resource type k in server i. Then, our goal is to select m disjoint subsets of VMs so that the total profit of the selected item is a maximum, and each subset can be assigned to a different server whose capacity of each resource is no less than the total capacity of corresponding resources of items in the subset. We formula this problem as the multidimension knapsack problem (MKP) [13]. Formally,

$$\max Z = \sum_{i=1}^{N}\sum_{j=1}^{M} p_{ij}X_{ij} \quad (5)$$

$$\text{s.t.} \quad \sum_{j=1}^{M} R_{jk}X_{ij} \leq T_{ik}, \forall i,k \quad (6)$$

$$\sum_{j=1}^{M} X_{ij} \leq 1, \forall\, i \quad (7)$$

$$X_{ij} = \{0,1\}, \forall\, i,k, \quad (8)$$
$$\forall i = 1,...,N, \forall j = 1,...,M$$

Constraint (6) ensures that total resource type k of each subset cannot exceed the capacity resource type k of the container. Constraint (7) allows at most one item j to be allocated to container k. This problem is related to knapsack problem and each of the k constraints described in (6) is called knapsack constraint. So our problem is also called the multidimensional multiple knapsack problem (MMKP) [10].

IV. Dynamic resource allocation in NFVO

In this chapter, we investigate three algorithms to allocate resource dynamically, implemented in Optimization module and Analysis and Prediction module.

A. Prediction server status

To improve the accuracy of the consolidation, the prediction algorithm can show the resource utility status on each server in the next time slot. From this information, we can know which server will be overloaded, or underloaded. With overloaded servers in the next time slot, we should not create a new VM or choose to be destinations in migrations. And, with underloaded servers, we should migrate all VMs out of them then turn to sleep mode. Our model uses the TCP-like scheme formula for prediction resources [7]. To calculate the prediction resource utility in the next time slot, we use the following formula:

$$\overline{r}(t) = \alpha * \overline{r}(t-1) + (1-\alpha) * O(t), 0 \leq \alpha \leq 1 \quad (9)$$

where $\overline{r}(t)$ is the estimated resource load and $O(t)$ is the observed resource load at time t.

Note that, this algorithm also reduces number of servers and numbers of VMs that are inputted into Optimization module.

B. Creating a new VM

Allocating resource for a new VM is a simple allocation scenario. However, to reduce the migration in the future, we must assign the new VM to the best server in term of price. Therefore, we apply the Best Fit algorithm, which tries to place the VM in the "tightest spaces" [13]. That means the chosen server must have the lowest price $pr^{j}_{service}(t)$ of this instance, but it still has an available slot for this instance. Furthermore, it will not be a hot spot in the next time slot. If there are many servers that have the same price, we will choose the server that has the lowest resource utility. This step will make our system load balance for serving requests.

Algorithm 1: Best Fit algorithm for GSC

Function *BestFit* **is**
  Input: The instance of VM will be created and a list prices of servers
  Result: The best candidate server
  $min := 1$;
  $pr_{min} \leftarrow$ price of server min;
  for $i \leftarrow 2$ **to** N **do**
    Finding a lowest price of input instance;
    if *(pr_{min} > the price of server i)* **then**
      $min \leftarrow i$;
      $pr_{min} \leftarrow$ price of server i;
    end
  end
end

C. Consolidation system by GA

In this part, we solve the optimization problem MMKP by GA. Many methods were proposed for solving this problem. However, with multi dimensions and multi knapsacks with difference size, the complexity is so high. As we mentioned above, in this paper we apply the GA from [10]. The authors already compared to other well-known heuristic methods and represented that this is a significant improvement in solution quality which the GA made over the other heuristics, whilst requiring only modest computing efforts, favors the choice of the GA. During the course of evolution, natural population evolves according to the principles of natural selection and "survival of the fittest" [10] (i.e., survival individuals are the candidates of optimal solution). The survival individuals have the genes which highly adapt to the environment condition (constraints). The combination of good characteristics from highly adapted ancestors may produce even more fit offspring. In this way, species evolve to become more and more well adapted to their environment. In GAs, there are three basic operators: *reproduction, crossover* and *mutation*, act on a population of candidate solution (chromosomes) in the search space.

1) Representation scheme: In solving the optimal value by a genetic algorithm, we must take into account how to represent individuals in the GA population. We convert matrix X(N,M) into one dimension S(1,NxM). Hence, we used a NxM-bit binary string, where n is the number of servers and m is the number of VMs.

Similarly, the vector profit p of VMs is also converted into one dimension $p(1, N*M)$. Then our objective function can represent again following formula:

$$f(S) = \sum_{j=1}^{N*M} p_j S_j \quad (10)$$

2) Genetic Operation of MMKP: First, we generate two parents who will have children. The chosen parents must satisfy the constraints but maybe they are not the best fitness.

In practically, the chromosomes are two binary strings that are randomly created. Second, the crossover and mutation operators will create individuals to overcome the natural selection, who may will be the optimal individuals in the popularity. In uniform crossover two parents have a single child. Its bit string is also generated by 0 or 1 that comes from the parents. Once a child solution has been generated through crossover, a mutation procedure is performed randomly in some selected bits [10]. The rate of mutation in our model is set to be small (about 1 or 2 bits in each string). Third, the child solution generated by the crossover and mutation procedures many not be satisfied the constraints. In [10], they applied a heuristic algorithm for repairing operator. This step implies fixing something in genes to adapt the constraints of environment. Finally, the eliminating process will occur to delete any duplicate children and the individuals who have lowest fitness value. The GA for MMKP will be represented in Algorithm 2.

Algorithm 2: GA for the MMKP

Function *allocateVMs* **is**
  Input: A list of VMs and a list of servers
  Result: Allocate VMs into servers with maximum the total profit
  Step 1: set t:=0;
  Step 2: randomly generate the initial population $P(t) = \{S_1, ..., S_T\}$;
  Step 3: evaluate the fitness of the initial population: $P(t) = \{f(S_1), ..., f(S_N)\}$;
  Step 4: find S^*;
  Step 5: execute crossover operator and mutation operator to individuals;
  Step 6: generate the new population $P(t+1)$;
  Step 7: evaluate the fitness of new population $P(t+1)$;
  if *(the termination criteria are not met)* **then**
    $t := t + 1$;
    go to **Step 3**;
  end
end

3) Consolidation servers: For consolidating servers, this is often a periodic task which is configured by an administrator. However, in some cases, when there are many underloaded servers, greater than the threshold of $Coldserver$, the Determination module can call this function to rearrange the whole system.

V. Simulation and numerical

In this section, we implement our model and evaluate efficient of our system in the context of realistic workloads of web servers.

Simulation environment: We consider a scenario that the system has maximum 100 active servers and create randomly 100 to 500 VMs from three instances types, which are referred

Algorithm 3: Consolidating servers

Input: X, S, P, Mi
// X: allocation matrix of VMs
// S: list of servers
// pr: price matrix of VMs
// p: profit matrix of VMs
// Mi: migration list
// Rl: List of releasing servers
Result: The list of migrations, list of releasing servers
Predicthotspot(S);
Filtering candidate servers for consolidation;
Updating X and S;
pr←Calculating prices of instances;
p← Calculating profit of VMs;
// Run the genetic algorithm for MMKP
allocateVMs(X,S);
Mi← get list of migration;
Rl← get list of release servers;
pr← Updating price of VMs;

TABLE I: Thresholds and parameters for consolidating server

Name	Value
Hot threshold	70%
Warm threshold	60%
Cold threshold	20%
Number of migration VMs (Mig_{thr})	20%
Number of cold spots ($Coldserver$)	20%
Population size	200
Crossover probability	0.4
Mutation probability	0.012

in [14]. With the mutation rate 0.012, crossover rate 0.4 and limitation of population size 200, we consider that our system can get convergence point within 400 iterations.

In other words, we apply cold threshold, warm threshold, and hot threshold as represented in Table I. We modify the values of these thresholds from [7] for adapting our algorithm. These thresholds are also used to prevent a VM get moved to a hot destination. If the hot threshold is higher than 70%, for a long time, many servers will become the overloaded servers. Moreover, if we increase the hot threshold, the system will lost the load balancing feature that plays an importance role in data centers.

In order to evaluate the combination of these algorithms, we also simulate the consolidation with bin-packing algorithm (such as Best Fit Decreasing) to solve the problem (5) then compare to our algorithm. Without considering thresholds in Table I and predicting hot spots algorithm, number of hot servers increase gradually. Since, the Best Fit Decreasing algorithm tries to migrate VMs into "the tightest hole" that makes the system unbalance. The more meaningful of these thresholds and the prediction algorithm is shown in Figure 2.

Information of workload: To evaluate our model, we use log files from web servers, which traces from 8 servers and 50 VMs in average. From the log file, we obtain the average

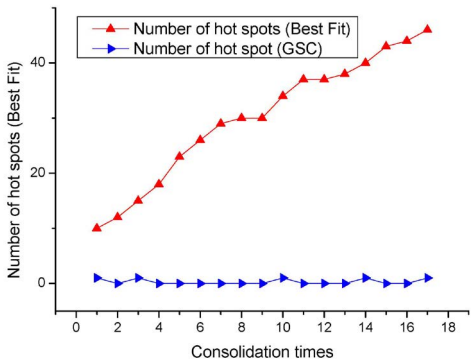

Fig. 2: In comparison with bin-packing algorithm.

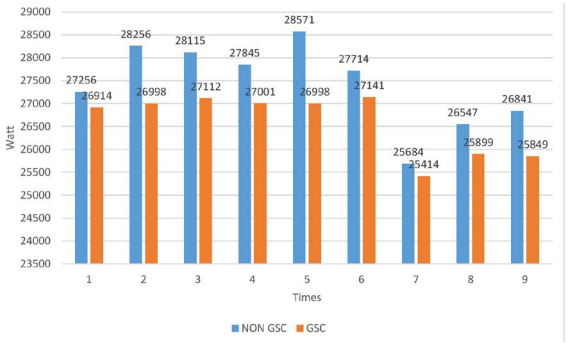

Fig. 3: Power consumption in GSC.

request rate 2000 requests per hour in this tracking system and trace the location of all VMs in 9 hours, starting at midnight on Monday July 1 2013. From 0 AM to 6 AM, the system has more underloaded servers than other periods. This time also is the best duration that we should consolidate the whole system. From the tracer file, we know information when a VM is created and remaining resources of each server. This information is used as input of our model. From the simulation, we collect essential performance indicators, the percentage of released servers and effects of saving power. We assume that each server is set 200 watts at idle and 400 watts at maximum performance. To evaluate the effect in reducing power consumption, we use the formula [15]:

$$Pw = (Pw_{\max} - Pw_{idle}) * \overline{r}_i + Pw_{idle}, \qquad (11)$$

where Pw_{max} is the power consumption at maximum performance and Pw_{idle} is the power consumption at idle.

The Figure 3 shows the effect of energy consumption by using GSC model. Comparing with without using GSC model, our model can save 1% to 3.7% power consumption. This reason is related to the number of released servers. We also investigate this algorithm when scaling the average resources utility into the hot threshold.

VI. Conclusion

In this paper, we propose a consolidation method in Cloud-NFV, to maximize the total profit and reduce the power consumption for servers in data centers. Server consolidation approach is considered as a placement problem in data centers. However, in multi dimension constraints on data centers, GA is one of the solutions to find out the optimal value. Although, we apply an improvement GA, still has high cost in computation, by using proposed parameters and some combined algorithms, GA can be a significant tool to find the optimal point in consolidation problem with the lowest gap. Especially, our work defines the elastic prices for serving VMs. This considering helps us not only to minimize the power consumption, but also to maximum the utilization on each servers. The analysis and simulation show that our proposed system is adaptable and can be applied to the scheduling function of data centers. As a future work, we want to apply widely our model in more practical environment, focus deeply on the live migration cost and the load balancing problem.

VII. Acknowledgement

This work was supported by the ICT R&D program of MSIP/IITP, Republic of Korea. (2014-044-011-003, Open control based on distributed mobile core network) *Dr. CS Hong is the corresponding author.

References

[1] Cloud NFV White Paper. [Online]. Available: http://cloudnfv.com/WhitePaper.pdf
[2] Cloud NFV Brochure. [Online]. Available: www.cloudnfv.com/Brochure.pdf
[3] ETSI NFV White Paper
. [Online]. Available: http://portal.etsi.org/NFV/NFV_White_Paper.pdf
[4] M. Uddin and A. A. Rahman, "Server consolidation: An approach to make data centers energy efficient and green," *arXiv preprint arXiv:1010.5037*, 2010.
[5] OpenStack Neat: A Framework for Dynamic Consolidation of Virtual Machines in OpenStack Clouds A Blueprint. [Online]. Available: http://www.cloudbus.org/reports/OpenStackneat-BlueprintAug2012.pdf
[6] T.-D. Nguyen, A. T. Nguyen, M. D. Nguyen, M. Van Nguyen, and E.-N. Huh, "An improvement of resource allocation for migration process in cloud environment," *The Computer Journal*, vol. 57, no. 2, pp. 308–318, 2014.
[7] Z. Xiao, W. Song, and Q. Chen, "Dynamic resource allocation using virtual machines for cloud computing environment," *Parallel and Distributed Systems, IEEE Transactions on*, vol. 24, no. 6, pp. 1107–1117, 2013.
[8] C. Mastroianni, M. Meo, and G. Papuzzo, "Probabilistic consolidation of virtual machines in self-organizing cloud data centers," 2013.
[9] M. Lin, A. Wierman, L. L. Andrew, and E. Thereska, "Online dynamic capacity provisioning in data centers," in *Communication, Control, and Computing (Allerton), 2011 49th Annual Allerton Conference on*. IEEE, 2011, pp. 1159–1163.
[10] P. C. Chu and J. E. Beasley, "A genetic algorithm for the multidimensional knapsack problem," *Journal of heuristics*, vol. 4, no. 1, pp. 63–86, 1998.
[11] E. Falkenauer and A. Delchambre, "A genetic algorithm for bin packing and line balancing," in *Robotics and Automation, 1992. Proceedings., 1992 IEEE International Conference on*. IEEE, 1992, pp. 1186–1192.
[12] A. Beloglazov and R. Buyya, "Openstack neat: A framework for dynamic consolidation of virtual machines in openstack clouds–a blueprint," *Cloud Computing and Distributed Systems (CLOUDS) Laboratory*, 2012.
[13] S. Martello and P. Toth, *Knapsack problems*. Wiley New York, 1990.
[14] [Online]. Available: http://aws.amazon.com/ec2/pricing/
[15] [Online]. Available: http://www.infoq.com/articles/power-consumption-servers