Green Containerized Service Consolidation in Cloud

Shubha Brata Nath*, Sourav Kanti Addya†, Sandip Chakraborty‡ and Soumya K Ghosh§
Department of Computer Science and Engineering
*‡§Indian Institute of Technology Kharagpur, India †National Institute of Technology Karnataka, Surathkal, India
Email: *nath.shubha@gmail.com, †souravkaddya@nitk.edu.in, ‡sandipc@cse.iitkgp.ac.in, §skg@cse.iitkgp.ac.in

***Abstract*—In the presence of latency sensitive geo-distributed applications, users require fast service for their queries. Cloud computing provides physical servers from its data center in order to process user requests. The cloud data center consumes a huge amount of energy due to lack of management of the data center servers as the container-based service consolidation is a non-trivial task. Since the containers require less resource footprint, consolidating it in servers might make resource availability sparse. In order to reduce the energy consumption of the cloud data center, we have proposed a green container-based consolidation of the services so that the maximum number of servers can be put into idle mode without affecting the application quality of experience. The service consolidation problem has been formulated as an optimization problem considering minimization of total energy consumption of the data center as the objective, and an algorithm named *Energy Aware Service consolidation using baYesian optimization* (EASY) has been proposed to solve the optimization. We have evaluated the EASY algorithm in simulation using python. The experimental results have shown that EASY improves the total energy consumption of the data centers. This improvement comes at the cost of a small increase of service response time as there exists a trade-off between energy consumption and service response time.**

***Index Terms*—Cloud Computing; Container; Service Consolidation; Energy Consumption; Bayesian Optimization**

I. Introduction

With the advent of latency sensitive geo-distributed applications like social media applications, video, and game streaming, etc., ensuring Quality of Experience (QoE) for the end users while minimizing the application response time is one of the prime requirements. In a typical cloud architecture [1], user applications are executed over virtual machines (VM) hosted over physical cloud servers, where resources from a large available pool are allocated to the VMs depending on the application QoE requirements. More recently, the industry is migrating towards container-based virtualization [2] as it provides a lightweight and easy-to-deploy environment for hosting end user services over a cloud data center. Further, containers have less resource footprint compared to VMs; therefore, it provides an additional advantage of supporting virtualization with less resource overhead.

Among various challenges of workload placements in a data center, minimizing the overall energy footprint is a major issue, as a typical data center consumes a significant amount of energy during the run time [3]. A major source of energy leakage from data center is the improper distribution of workloads across the data center servers, which reduces the server utilization per unit amount of energy consumption [4]. Such a situation occurs because the workloads are placed on-the-fly as and when they arrive, and this dynamic scheduling of workloads lead to a situation where a majority of the servers remain busy, although they may not be fully utilized. A number of recent researches have focused on dynamic VM placements over a data center considering the energy-aware scheduling of workloads, known as VM-based service consolidations [5], [6]. In comparison to the above, containerized service consolidations have additional challenges as follows. (i) Containers are loosely-coupled with the host operating system compared to VMs which are tightly-coupled and run a separate guest operating system. Therefore, it is difficult to monitor and predict the resource usage of the containerized workloads. Probing a containerized service for workload characterization introduces noise in the measurements [7]. (ii) Containers can be stateful or stateless depending on the workload type; for multi-tier workloads, containers are in general stateful. For stateful containers running multi-tier workloads, the inter-container coordination consumes additional resources which need to be considered for containerized service consolidation. (iii) Container migrations in general work in batch, where multiple containers are deployed or migrated based on the workload scheduling policy. The inter-dependency of containerized services need to be considered while developing the service consolidation methodology.

Owing to the above challenges, in this paper, we have proposed an approach for green container-based service consolidation in the cloud data centers in order to minimize the total energy footprint of the data center. This problem has been formulated as an optimization problem where the objective is to minimize the total energy consumption while maintaining the workload QoE. It can be shown that the problem of container-based service consolidation with the above objectives is $\mathcal{NP}$-hard as the bin packing problem is reducible to the container-based service consolidation problem. Therefore, we apply a statistical online learning based technique based on Bayesian Optimization (BO) [8]. The proposed technique is scalable in the presence of containerized service placements in batches. Further, BO is known to be robust against statistical noises introduced during observations, and therefore, the proposed technique can tolerate measurement noises introduced during workload characterization for containerized services. The proposed solution, called *Energy Aware Service consolidation using baYesian optimization* (EASY), has been implemented and tested over a realistic emulation scenario. We have also compared the performance of EASY with various state-of-the-

978-1-7281-5089-5/20/$31.00 ©2020 IEEE

art techniques [9]–[11] available in the literature. We have observed that EASY is able to outperform these algorithms in terms of the total energy consumption, while maintaining the service QoE.

The rest of the paper is organized as follows. We discuss the related works in Section II. Section III presents the system architecture used in this work. Section IV presents the mathematical modeling of the container-based service placement problem in a data center as an optimization problem. The BO based algorithm to solve the optimization problem is given in Section V. In Section VI, we give the evaluation of the proposed model. Finally, Section VII concludes the paper.

II. Related Works

The existing literature has works related to the consolidation of services as a VM in cloud data centers. Tziritas et al. [5] have proposed an algorithm in order to minimize the energy consumption within the system. This improvement comes at the cost of a small increase in terms of service level agreement (SLA) violations. The dynamic consolidation of a virtual machine is also done by using reinforcement learning in [12]. The work optimized the number of active servers according to the current resource utilization. The intelligent agent takes the decision about when to switch a host into active or inactive. The authors in [6], have used TD(λ)-learning algorithm which is a reinforcement learning algorithm to do a trade-off between the total energy consumption and average job response time in a data center.

In the literature, there are existing works related to the container-based service consolidation in cloud data centers. Lin et al. [9] have studied the minimization of energy consumption of container-based virtualization in a cloud data center. The problem has been solved by a greedy heuristic as well as a dynamic programming approach. In [13], the authors focused on the problem of energy management of the servers by proposing a framework to consolidate containers on VMs. The work does not consider the VM startup delay and migration overhead. Also, the host selection might not be optimal. The authors in [14] experimentally compared the performance of VMs and containers to show that the consolidation of containers perform better than the VM consolidation with respect to quality of service as well as energy. The limitation of the work is that it does not consider the costs of migrating containers. There are some works related to the workload consolidation on a minimum number of physical servers when the services are deployed as containers and virtual machines. In [15], the container consolidation problem has been solved by deep reinforcement learning. The limitation of the work is that the deep learning model needs much more amount of resources to find the solution. Shi et al. [16] have performed container consolidation in cloud using particle swarm optimization to reduce energy consumption. The container consolidation is done by placing the containers in VMs and placing the VMs in physical machines. Liu et al. [17] have performed container consolidation with usage prediction to improve SLA and energy efficiency in cloud data centers. The authors have used a linear regression model for this. The limitation is in the size of the data needed to do prediction.

There are some works in the literature that have performed container consolidation in cloud data centers without considering energy consumption. In [18], the deployment of micro-services using containers has been optimized in multi-cloud. The cloud service cost, network latency among the micro-services and micro-service restart time have been considered for optimization. The authors in [19] have studied different container networks. In this work, the overhead of container networks has been compared.

In conclusion, the usage of virtual machines in the works [5], [6], [12] makes services heavy-weight. Hence, placement and reassignment are time-consuming. Though used containers for service deployment, the greedy heuristic and the dynamic programming method used in the work [9] is not able to find a good solution. In order to find a better solution than the literature we have reviewed, a light-weight bayesian optimization based algorithm has been used to find a near-optimum solution.

III. Architecture of the System

We propose the architecture of the system, as shown in Figure 1, which has the following components.

1) **End Users:** End users are connected to the service layer by the internet. They send a request for service to the data center (DC) manager node residing in the service layer.
2) **DC Manager Node:** The DC manager node is a special node that orchestrates all the DC worker nodes present in the DC network. DC manager node has a BO based orchestrator module to run our proposed EASY algorithm and to place the services in the DC network based on the output of EASY. This node also aggregates the results obtained from the other DC worker nodes and sends the final result to the client node or the end users. A container engine is also running in this node to run different containers.
3) **DC Worker Node:** DC worker node runs the different containers to perform service execution. This node also has container engine for management of different containers. The DC manager node and the DC worker node constitute the cloud.
4) **Data Sources:** The data sources are connected with the cloud data center nodes by the internet. Different containers fetch the data from these data sources for processing.

Now, we present the mathematical modelling of the green containerized service consolidation problem using the above architecture in the next section.

IV. Modelling of the Service Placement in Data Center

Let us consider that there are n servers in the data center and m containers present in the system. We denote $N = \{N_i : i \in (1, \ldots, n)\}$ and $Cn = \{Cn_j : j \in (1, \ldots, m)\}$ as sets of DC servers and containers respectively.

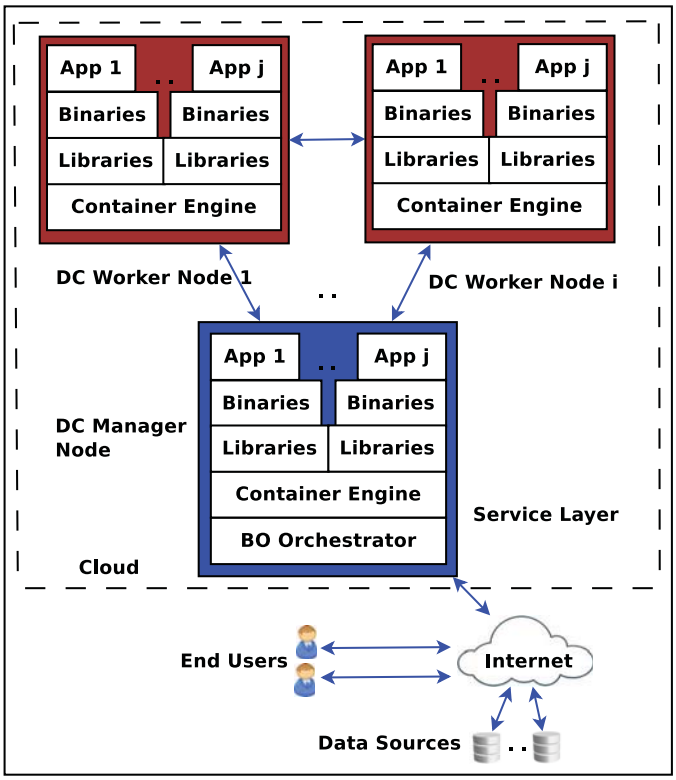

Fig. 1. System architecture

A. DC Worker's Energy Consumption

We define the total energy consumption of a server as

$$E_{N_i} = E_{N_i}^{rcv-data} + E_{N_i}^{rcv-frm-manager} + (1-\alpha+\beta)\times E_{N_i}^{comp} + \beta \times (E_{N_i}^{rcv-frm-node} + E_{N_i}^{send-to-node}) + \alpha \times (E_{N_i}^{offl-node} + E_{N_i}^{rcv-frm-node}) + E_{N_i}^{send-to-client}$$

where α ($0 \leq \alpha < 1$) and β ($0 \leq \beta < 1$) are percentage of data offloaded to another server (worker) and percentage of data received from another server (worker) respectively.

The sending or receiving of bits in a server consumes energy. It is defined as follows:

$$E_{N_i}^{offl-node} = E_{N_i}^{send} = E_{N_i}^{rcv} = E_{bit} \times B$$

where E_{bit} is the energy consumed to transfer one bit of data and B is the number of bits send or received by the server.

In the data center, the main source of power consumption is the central processing unit (CPU) [20] of the server. The energy consumed for computation is defined as follows [20]:

$$E_{N_i}^{comp} = (E_{N_i}^{max-comp} - E_{N_i}^{idle-comp}) \times U_{N_i}^{CPU} + E_{N_i}^{idle-comp}$$

where $U_{N_i}^{CPU}$ is the CPU utilization of the server N_i.

B. Constraints

The energy consumed by a node in a data center has an upper bound. We define this as follows: $E_{N_i} \leq \mathcal{E}_{N_i}$ where $\mathcal{E}_{N_i}$ is the upper limit of the energy consumption of a DC server N_i. The maximum energy consumption of a DC server happens when the CPU utilization of the server N_i is maximum i.e. $Umax_{N_i}^{CPU}$.

Let N_i^{CPU}, N_i^{RAM}, and N_i^{BW} be the available CPU, available random-access memory (RAM), and available bandwidth of the server N_i respectively. The resources needed by the container Cn_j are Cn_j^{CPU}, Cn_j^{RAM}, and Cn_j^{BW} respectively. We need to find a subset of p servers (N_1,..., N_p) from the set of all servers N where $\mathcal{P} \subseteq N$ such that the following conditions are true.

$$\sum_{j=1}^{TCn_i} Cn_j^{CPU} \times x_{i,j} \leq N_i^{CPU} \times y_i$$

$$\sum_{j=1}^{TCn_i} Cn_j^{RAM} \times x_{i,j} \leq N_i^{RAM} \times y_i$$

$$\sum_{j=1}^{TCn_i} Cn_j^{BW} \times x_{i,j} \leq N_i^{BW} \times y_i$$

where i = 1,...,n; $0 \leq j \leq m$ and we define $x_{i,j}$ as the decision variable to denote if a container Cn_j has been placed in the server N_i. Also, y_i is the decision variable to denote if a server N_i is used for placement or not. TCn_i is the total number of containers present in the server N_i. It is the sum of the containers from the current server's execution and the containers offloaded by other server into this server.

$$p = \sum_{i=1}^{n} y_i \geq 1; \underset{i,j}{\forall} x_{i,j} \in \{0,1\}; \underset{i}{\forall} y_i \in \{0,1\}$$

The total number of containers in all the servers should be equal to the total number of containers present in the system.

$$\sum_{i=1}^{p} TCn_i = |Cn|$$

where $|Cn|$ is the cardinality of the set Cn.

The following equation states that one container can be placed in at most one server.

$$\underset{j}{\forall} \sum_{i=1}^{p} x_{i,j} = 1$$

The CPU utilization of the server [21] N_i is defined as:

$$U_{N_i}^{CPU} = \frac{\sum_{j=1}^{TCn_i} Cn_j^{CPU}}{N_i^{CPU}} \leq Umax_{N_i}^{CPU}$$

C. Problem Definition

We present the formal definition of the green containerized service consolidation problem of a cloud data center as follows.

Given a set of containers (Cn) with resource requirement and a set of physical servers as workers (N) with resource availability, our objective is to place the containers in an optimum number of servers (p) so that the total energy consumption ($\sum_{i=1}^{p} E_{N_i}$) is minimized without affecting the

application quality of experience. Therefore, mathematically the objective function can be represented as Equation (1).

$$\text{minimize} \quad \sum_{i=1}^{p} E_{N_i} \tag{1}$$

subject to the constraints mentioned in IV-B.

It can be shown that the bin packing problem [15], a known $\mathcal{NP}$-Hard problem, is polynomial time reducible to our green containerized service consolidation problem given in Equation (1). Therefore, the problem of green containerized service consolidation in the cloud data center is $\mathcal{NP}$-Hard. However, we exclude this proof due to the space constraints of this paper.

D. Relationship between the Parameters

With the increase of the number of active servers i.e. $\sum_{i=1}^{n} y_i$, the total energy consumption of a data center ($\sum_{i=1}^{p} E_{N_i}$) increases. However, with the increase of number of active servers ($\sum_{i=1}^{n} y_i$), the average response time decreases.

V. Algorithm Design

As the problem of container-based service consolidation in the cloud data center is $\mathcal{NP}$-Hard, we need a heuristic to get a near-optimal solution. We have used BO to solve this problem.

A. Why Bayesian Optimization?

The motivations behind choosing BO to solve this problem are stated as follows. (i) BO is a technique to optimize black-box functions for which no closed form is known (nor its gradients). This is because BO does not need the function to be parametric. (ii) BO is applicable to the functions that are expensive to evaluate. BO needs a very small number of sample points to reach a near-optimal solution. (iii) BO can be applied to the functions whose evaluations are noisy. BO uses the confidence interval to find the solutions with some guarantee in the presence of noise.

B. Solution of the Optimization

Let us consider that the objective function $E_{worker}(\mathcal{P}) = \sum_{i=1}^{p} E_{N_i}$ follows a gaussian distribution. BO uses acquisition function (EI) to find the configuration in an iteration. We choose the acquisition function (EI) based on [22]. But, BO works well in case of unconstrained optimization. In order to satisfy our constrained optimization, we have modified EI by using [23]. It is given as follows:

$$EI^c(\mathcal{P}) = P(\mathcal{C})\, EI(\mathcal{P})$$

where $\mathcal{C}$ is the set of constraint values. Also, $\mathcal{C}$ follows Bernoulli process.

Algorithm 1: EASY Procedure

Input: Container Set Cn, DC Server Set N
Output: The configuration ($conf\_m$) for which the total energy consumption of the data center $\sum_{i=1}^{p} E_{N_i}$ is minimum

```
Function Main():
    Initialize two configurations conf_1 and conf_2;
    /* Conf is the configuration set */
    Conf = conf_1 ∪ conf_2 ;
    for (i = 0; HasConverged()! = True; i = i + 1) do
        Calculate expected improvement by calling EI^c(P);
        Choose the next configuration (conf_i) based on the expected improvement values;
        Conf = Conf ∪ conf_i;
    conf_m = Find the configuration from the Conf set for which the total energy consumption of the data center ∑_{i=1}^{p} E_{N_i} is minimum;
    return conf_m;
```

C. EASY Algorithm

The inputs to the algorithm are the set of containers to be placed and set of data center server. The algorithm produces as output the configuration ($conf\_m$) for which the total energy consumption of the data center $\sum_{i=1}^{p} E_{N_i}$ is minimum. The proposed EASY algorithm is described in Algorithm 1.

VI. Performance Evaluation

To study the run time behavior of the EASY algorithm, we performed the experiments in simulation using python 2.7.13. We have used skopt (https://scikit-optimize.github.io/) python library to implement BO algorithm. We have done profiling of docker (https://www.docker.com/) containers offline and used those values in the python based BO simulation. Table I shows the values of the simulation parameters taken.

TABLE I
Simulation Parameters

Parameter	Value
Number of workers	100
Number of containers	2000 - 3050
Maximum available memory of a worker	16 GB, 32 GB, 64 GB
Required memory of a container	450 MB - 2178 MB
Total data size to be processed	2600 MB - 10400 MB
Percentage of data offloaded to another server(α)	0.3, 0.7
Percentage of data received from another server(β)	0.3, 0.7

A. Competing Heuristics

We consider *Consecutive Allocation* [9], *Best Fit* [10], and *First Fit Decreasing* [11] mechanisms as the baseline methods to analyze the performance of EASY. In *Consecutive Allocation* heuristic, the input container sequence is divided

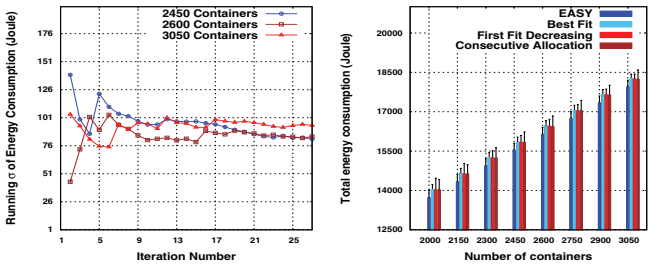

Fig. 2. (a) Convergence of EASY Algorithm and (b) Performance in terms of Total energy consumption of the workers

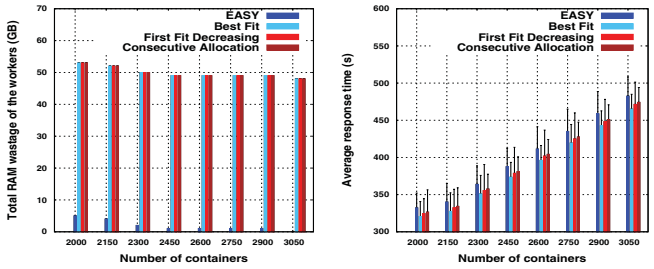

Fig. 3. (a) Performance in terms of Total RAM wastage of the workers and (b) Performance in terms of Average response time

into disjoint sets having a number of containers. The containers that are present in a set are deployed to the same server. The physical server that has the smallest sufficient resource is chosen in the *Best Fit* mechanism. In the *First Fit Decreasing* algorithm, the containers are sorted in decreasing order of their resource requirement. The sorted containers are placed to the first available server.

B. Convergence of the EASY Algorithm

We have tested the convergence of EASY algorithm for different scenarios and the results of convergence are shown in Figure 2(a). For this analysis, we have taken the running standard deviation (σ) of energy consumption values given by the EASY algorithm. It is observed that the EASY algorithm converges within 20 iterations for a different number of containers to be placed in the cloud data center. We have taken this value as the value for the number of iterations for different scenarios.

C. Total Energy Consumption of the Worker Nodes

Figure 2(b) shows the total energy consumption of the active servers (workers) with the increase of the number of containers to be placed. It is found that the proposed EASY algorithm consumes less amount of energy with respect to best fit, first fit decreasing algorithm as well as consecutive allocation algorithm as the EASY algorithm finds the allocation for which the energy consumption would be minimal.

D. Total RAM wastage of the Worker Nodes

The effect of the increase in the number of containers on the total RAM wastage is shown in Figure 3(a). It can be noted that the total RAM wastage of the workers is less for EASY as it is minimizing the number of active workers by placing the containers on them.

E. Average Response Time of the Containers

The EASY algorithm does placement based on the minimum value of the energy consumption. But, the average response time is not less than the baseline allocations as the decrease in the number of active workers make them heavy loaded. Thus, the average response time suffers a little. Figure 3(b) shows the effect on the average response time of the containers.

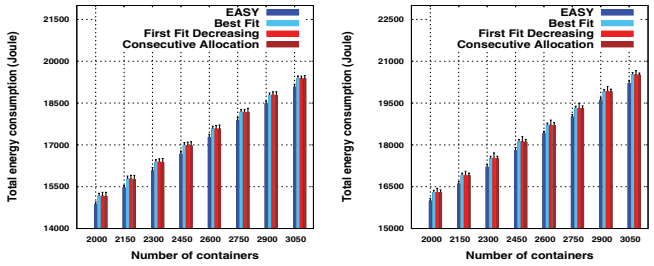

Fig. 4. (a) Total energy of the workers when the data size is 5200 MB and (b) Total energy of the workers when the data size is 10400 MB

F. Increase of Data Size to be Processed

We have seen that the total energy consumption of the data center worker nodes increases with the increase of data size to be processed. The result for the data size of 5200 MB is shown in Figure 4(a) and the result for the data size of 10400 MB is shown in Figure 4(b). This is happening because the receiving of more amount of data to be processed increases the energy consumption. We have observed that the EASY algorithm performs better than the baselines in these cases also.

G. Increase of Container Migration

The performance of the EASY algorithm is analyzed under different container migration scenarios. We have observed that the energy consumption increases with the increase of container migrations. The energy consumption increases as the worker nodes receive more number of containers. Our EASY algorithm also performs better than best fit, first fit decreasing as well as consecutive allocation algorithm as BO based algorithm is intelligent enough to choose the near-optimum solution based on the prior observations. The percentage of data offloaded to another worker (α) and the percentage of data received from another worker (β) are taken as 0.3 in Figure 5(a). The parameters are taken as 0.7 in Figure 5(b).

H. Overhead of the Data Center Manager Node

The manager node consumes energy due to the communication with the client nodes and the worker nodes. Also, the manager node consumes energy as it runs the proposed EASY algorithm for orchestration of the services. The total energy consumption of the manager node is compared with the baselines and shown in Figure 6(a). We have also found out

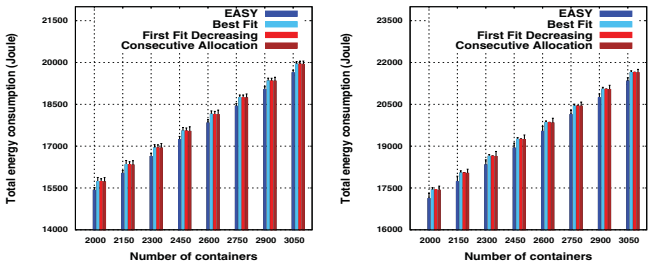

Fig. 5. (a) Total energy of the workers when α = 0.3, β = 0.3 and (b) Total energy of the workers when α = 0.7, β = 0.7

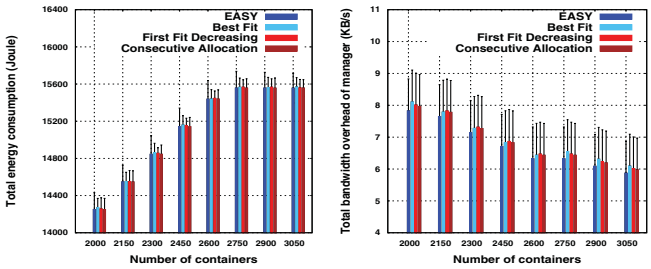

Fig. 6. (a) Total energy of the manager node and (b) Total bandwidth overhead of the manager node

the overhead of the manager node in terms of the bandwidth usage and shown in Figure 6(b).

VII. Conclusion

We have presented an approach for the green container-based service consolidation to reduce the energy consumption of the cloud data center. There exists a trade-off between the service response time and total energy consumption of the data center. We have chosen the minimization of total energy consumption over minimization of service response time. A bayesian optimization based container consolidation algorithm named EASY has been proposed for this. To study the run time behavior of the problem, we performed the experiments in simulation. It is found that the proposed EASY algorithm consumes less amount of energy with respect to the baseline algorithms as our EASY algorithm finds the placement for which the total energy consumption would be minimal. It is also observed that the total memory wastage of the servers is minimum for EASY as it is minimizing the number of active servers. But, the average response time is a little more than baseline algorithms as the decrease of the number of active servers make them heavy loaded. We have analyzed the overhead of the data center manager node for the proposed algorithm and the baseline algorithms. In the future, we are planning to evaluate EASY in a multi-cloud environment.

References

[1] P. Mell, T. Grance *et al.*, "The nist definition of cloud computing," 2011.

[2] M. A. Rodriguez and R. Buyya, "Container-based cluster orchestration systems: A taxonomy and future directions," *Software: Practice and Experience*, 2018.

[3] J. W. Smith, A. Khajeh-Hosseini, J. S. Ward, and I. Sommerville, "Cloudmonitor: profiling power usage," in *Proc. of the 5th IEEE CLOUD*. IEEE, 2012, pp. 947–948.

[4] M. Dayarathna, Y. Wen, and R. Fan, "Data center energy consumption modeling: A survey," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 732–794, 2015.

[5] N. Tziritas, S. Mustafa, M. Koziri, T. Loukopoulos, S. U. Khan, C.-Z. Xu, and A. Y. Zomaya, "Server consolidation in cloud computing," in *Proc. of the 24th IEEE ICPADS*. IEEE, 2018, pp. 194–203.

[6] X. Lin, Y. Wang, and M. Pedram, "A reinforcement learning-based power management framework for green computing data centers," in *Proc. of the IEEE IC2E*. IEEE, 2016, pp. 135–138.

[7] A. Anwar, M. Mohamed, V. Tarasov, M. Littley, L. Rupprecht, Y. Cheng, N. Zhao, D. Skourtis, A. S. Warke, H. Ludwig *et al.*, "Improving docker registry design based on production workload analysis," in *Proc. of the 16th USENIX FAST*, 2018, pp. 265–278.

[8] J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms," in *Proc. of the 25th NIPS*, 2012, pp. 2951–2959.

[9] C.-C. Lin, J.-J. Chen, P. Liu, and J.-J. Wu, "Energy-efficient core allocation and deployment for container-based virtualization," in *Proc. of the 24th IEEE ICPADS*. IEEE, 2018, pp. 93–101.

[10] A. Verma, P. Ahuja, and A. Neogi, "pmapper: power and migration cost aware application placement in virtualized systems," in *ACM/IFIP/USENIX International Conference on Distributed Systems Platforms and Open Distributed Processing*. Springer, 2008, pp. 243–264.

[11] J. Xu and J. A. Fortes, "Multi-objective virtual machine placement in virtualized data center environments," in *2010 IEEE/ACM Int'l Conference on Green Computing and Communications & Int'l Conference on Cyber, Physical and Social Computing*. IEEE, 2010, pp. 179–188.

[12] F. Farahnakian, P. Liljeberg, and J. Plosila, "Energy-efficient virtual machines consolidation in cloud data centers using reinforcement learning," in *2014 22nd Euromicro International Conference on Parallel, Distributed, and Network-Based Processing*. IEEE, 2014, pp. 500–507.

[13] S. F. Piraghaj, A. V. Dastjerdi, R. N. Calheiros, and R. Buyya, "A framework and algorithm for energy efficient container consolidation in cloud data centers," in *Proc. of the IEEE DSDIS*. IEEE, 2015, pp. 368–375.

[14] I. Cuadrado-Cordero, A.-C. Orgerie, and J.-M. Menaud, "Comparative experimental analysis of the quality-of-service and energy-efficiency of vms and containers' consolidation for cloud applications," in *Proc. of the 25th SoftCOM*. IEEE, 2017, pp. 1–6.

[15] S. Nanda and T. J. Hacker, "Racc: resource-aware container consolidation using a deep learning approach," in *Proceedings of the First Workshop on MLCS*, 2018, pp. 1–5.

[16] T. Shi, H. Ma, and G. Chen, "Energy-aware container consolidation based on pso in cloud data centers," in *Proc. of the IEEE CEC*. IEEE, 2018, pp. 1–8.

[17] J. Liu, S. Wang, A. Zhou, J. Xu, and F. Yang, "Sla-driven container consolidation with usage prediction for green cloud computing," *Frontiers of Computer Science*, vol. 14, no. 1, pp. 42–52, 2020.

[18] C. Guerrero, I. Lera, and C. Juiz, "Resource optimization of container orchestration: a case study in multi-cloud microservices-based applications," *The Journal of Supercomputing*, vol. 74, no. 7, pp. 2956–2983, 2018.

[19] K. Suo, Y. Zhao, W. Chen, and J. Rao, "An analysis and empirical study of container networks," in *Proc. of the IEEE INFOCOM*. IEEE, 2018, pp. 189–197.

[20] M. H. Ferdaus, M. Murshed, R. N. Calheiros, and R. Buyya, "Virtual machine consolidation in cloud data centers using aco metaheuristic," in *European conference on parallel processing*. Springer, 2014, pp. 306–317.

[21] S. K. Addya, A. K. Turuk, B. Sahoo, M. Sarkar, and S. K. Biswash, "Simulated annealing based vm placement strategy to maximize the profit for cloud service providers," *Engineering Science and Technology, an International Journal*, vol. 20, no. 4, pp. 1249–1259, 2017.

[22] O. Alipourfard, H. H. Liu, J. Chen, S. Venkataraman, M. Yu, and M. Zhang, "Cherrypick: Adaptively unearthing the best cloud configurations for big data analytics." in *Proc. of the 14th USENIX NSDI*, 2017, pp. 469–482.

[23] J. R. Gardner, M. J. Kusner, Z. E. Xu, K. Q. Weinberger, and J. P. Cunningham, "Bayesian optimization with inequality constraints." in *ICML*, 2014, pp. 937–945.