Efficient Cloud Resource Scheduling for Stochastic Demand with Heterogeneous Cost Models

Wei Wei
College of Information Science and Engineering
Henan University of Technology
Zhengzhou, China
nsyncw@126.com

Yang Liu
College of Information Science and Engineering
Henan University of Technology
Zhengzhou, China
enjoyang@126.com

***Abstract*—Distributed cloud platforms facilitate service providers to deliver geographically dispersed online services to a large number of users all over the world, while the aggregated user requests introduce stochastic demands for various resources in different cloud data centers. Resource scheduling in cloud platform is of high complexity due to various nonlinear cost models and multidimensional demand stochasticity. Existed scheduling algorithms generally utilize linear cost model, thus is hard to utilize budget efficiently. We proposed an efficient Heterogeneous Cost models oriented cloud Resource Scheduling (HCRS) to address the problem. Experiments results show that HCRS increase revenue by up to 40% than that in previous mean demand based algorithm and is of low enough complexity. Therefore, HCRS can be used as a candidate for scheduling in cloud platforms with heterogeneous cost models.**

Keywords—heterogeneous cost models; demand stochasticity; cloud computing; resource scheduling

I. INTRODUCTION

Globalized online services (such as Netflix [1], Facebook [2] and YouTube [3]) often face vast amounts of user requests from different locations across the world. Cloud computing technology is utilized to server user requests and reduces cost. Provided with nearly unlimited resources in cloud, under given resource budget, one common task of service provider is scheduling resources across geographically dispersed cloud data centers, to serve as many requests as possible while satisfy given Quality of Service (QoS) requirements. Usually, a nearer data center is expected to provide higher QoS. Although cloud computing platform can provide resources on demand, the delay between request and response varies for different resource kind, e.g., it takes nearly tens of minutes to setup and start a new Virtual Machine (VM) to fulfill new computing resource demand. As a result, the provided computing resources cannot precisely match immediate resource demand. The situation of resource insufficiency and overload become worse when handling high dynamic resource demands with time. How to alleviate above situation when pre-allocating resources is an impending need to meet in cloud platform. Moreover, since relative high response time is often a must under time-tense online scheduling, it is essential to compute placement efficiently.

When handling high dynamic resource demands, stochastic demand model is proved to be more effective than mean demand model, where demand is represented as a distribution in the former model and mean value in the latter model. Intuitively, stochastic demand model is effective since it can utilize more detail information of user demand than mean demand model. However, stochastic demands will significantly increase computation complexity. It is challenging to efficiently solve a resource allocation problem that combines combinatorial aspects and arbitrary stochastic demand distribution. Some preliminary research work have been carried out in stochastic demand based scheduling [4], [5], [6]. But existed work only investigate problems under amount constraint, e.g., maximize revenue for given amount of resources. Cost budget is more suitable for cloud resource scheduling since in typical cloud platform, resources usually incur rent cost. But the cost varies with instance type and data center location, and may change per several minutes according to the nonlinear charging functions. Cost budget is more general for scheduling when service providers rent resource in realistic cloud platforms. But charging functions in each data center of each cloud provider like [7] and [8] maybe different, the nonlinear stochastic demand combining with nonlinear charging functions make scheduling problem highly sophisticated.

To solve scheduling problem effectively and efficiently under this problem schema, one need to utilize the inner structure of the problem. We develop an efficient placement algorithm accordingly, suitable for time-tense scheduling in large-scale systems. We proved that the lower bound of target problem can be calculated using solution to original problem with amount constraint, which can be solved efficiently. We then develop an Heterogeneous Cost models oriented cloud Resource Scheduling algorithm (called HCRS), which starts from solution giving lower bound and gets improved solution quickly by leveraging discrete functions to approximate continuous ones. Under the scenarios with general settings, we compared the effect of HCRS with commonly used mean demand based algorithm. Our analysis and preliminary experiments indicate that HCRS outperforms mean demand based algorithm under all scenarios and can increase revenue by up to 42%. Therefore, it can be used as an effective supplement to existing algorithm under cost constraint scheduling scenarios.

II. Related Work

The first part of related research effort being made is of handling resource scheduling for multiple cloud based services [9], [10], [11], [12], [13], [14]. [9] forecasted the mean value of total streaming demands in given time span t, then provided optimal solutions to schedule cloud servers with diverse charging functions and capacities. Their solution also considered in location heterogeneity to mitigate the impact of user globalization. [10] utilized the queuing theory to investigate the cloud resource allocation problem. A queuing model is introduced to characterize the service process in multimedia cloud. The mean value of average request arrival rate is forecasted to help model user requests as a Poisson process. [11] aimed to schedule cloud resources to meet Service Level Agreement (SLA) for Video on Demand (VoD) applications at a modest cost, they use mean value to represent the bandwidth of users and user groups, then presented a distributed heuristic algorithm to get a budget solution that get scheduling plan at modest cost. [13] connected problem of bandwidth allocation with differential pricing, then proposed a two-phase flexible bandwidth allocator (A-FBA), which admits and allocates minimal bandwidth to dynamically serve arriving user requests in one phrase, and allocates additional bandwidth for accepted requests maximizing revenue in another phase.

There were also substantial research efforts being made about similar problems of resource consolidation [15], [16]. Based on a Markov chain model, [15] tried to detect host overload through maximizing the average intermigration time under given QoS goal, then adapted the algorithm to handle unknown non-stationary workloads using the multi-size sliding window workload estimation. [16] treated virtual machine as moldable unit which can change capacity when consolidation, then developed a Genetic Algorithm (GA) to consolidate mold-able VMs. Different from mean demands related work above, stochastic demand is considered to get optimal solution [17], [18]. [17] forecasted the distribution of demand and exploited the correlation between these demands. They used statistical multiplexing to reduce the probability of resource under-provision. [18] used user distribution to profile applications on given server. The target is to determine the degree of overbooking to guarantee requested performance while try to hold as many applications as possible.

The part of highly relevant work is that of stochastic demand based resource scheduling for geographically dispersed services [4], [5], [19], [20]. [4] tried to solve the resource placement problem in a geographically dispersed system which must satisfy varying demands for various kinds of resources. The algorithm aimed at placing resources in the regions as to maximize satisfied demand under resource constraints of dedicated data center in each region. In optimization target the local and remote requests are given different weight, thus can help consider in QoS and the revenue of resource sharing among data centers. [5] mitigated the problem in [4] to cloud platform where no resource constraints existed in each data center. The resource cost in considered in scheduling and the target is to allocate resource until the revenue by satisfying user demand is lower than resource cost. [19] proposed a scheduling algorithm for live media streaming system in cloud. They also considered stochastic demand and convert the resource scheduling problem to a min-cost flow problem, and then used a variant of successive shortest path to solve it quickly. [20] considered the problem of distributing resources to cloud data centers, to satisfy multidimensional and stochastic demands. The resource constraint is represented as a global resource amount budget. They convert the original problem into two nested optimization sub-problems, then derived an efficient fast resource placement algorithm. Existed work discussed the stochastic demand oriented resource scheduling problem in detail, and QoS and resource sharing are also involved to get an optimal result, while more realistic factors are not considered due to complexity is already very high when dealing with stochastic demand. However, to fit the algorithm to realistic scenario in cloud, the cost related effect should also be considered in when dealing with resource scheduling, which will be investigated in the paper.

III. Problem Formulation

We consider a general system consisting of J regions indexed by $j(1 \le j \le J)$. In each region, the service provider can place resources to satisfy local and remote requests. To fulfill the needs of realistic systems, we assume two different time scales $T_d < T_s$, where T_d (the measurement duration) stands for the duration time of measurement for stochastic demand in each region (e.g., ten minutes for stochastic demand measurement in media streaming system [17]), and T_s (the scheduling duration) represents the duration time of one charging period and is on the order of hours (e.g., the charging period of Amazon EC2 is one hour [7]). Since one T_s can cover several T_d, we set $T_s = IT_d$.

In one scheduling plan, we denote L_j as the amount of resource placed in region j, the set of resources placed in all regions is called a *placement* and the resource amount is denoted by $P = \{L_j\}$. And $L = \sum_{j=1}^{J} L_j$ as the total amount of resource. To be consistent to the model of cloud computing, we set no limit on the amount of resources in each region. Given a placement, for a request from region j, if it is accepted, then the request is labeled as '*satisfied*'. If a request from region j is satisfied, it is assigned to either of: 1) resources located in region j (satisfied locally), or 2) resource located in a different region (satisfied remotely). In each region, resources incur cost and the pricing model of each region is different. Denote the pricing model of each region as $f_j^c(\cdot)$, and the only assumption is that $f_j^c(\cdot)$ is a monotonically increasing function. The problem can be stated as how to maximize revenue under given budget. To get the closed form of the problem, we first introduce the revenue model.

Revenue Model. Given a placement P, in one realization of stochastic demand, let $S^{loc}(P)$ and $S^{rem}(P)$ denote the number of requests satisfied locally and remotely, then the weighted summation of satisfied demand is:

$$R(P) = w_{loc} S^{loc}(P) + w_{rem} S^{rem}(P). \quad (1)$$

Denote $S^{sat}(P)$ as the total number of satisfied requests, there is $S^{sat}(P) = S^{loc}(P) + S^{rem}(P)$, then (1) can be rewritten as:

$$R(P) = w_{loc} S^{loc}(P) + w_{sat} S^{sat}(P), \quad (2)$$

where w_{loc}, w_{sat} and w_{rem} are weights. For stochastic demand, let D_j^i be a random variable denoting the stochastic demand in the i -th measurement duration in region j, and D^i be a random variable denoting the demand in the i -th measurement duration in all regions. Their CDFs (Cumulative Distribution Function) are denoted as cdf_j^i and cdf^i, respectively. Given demand realization d_j^i and d^i, in the i -th measurement duration , the satisfied demand in region j is $min(d_j^i, L_j)$ the satisfied demand in all regions is $min(d^i, L)$. By concretize (2), for one realization of stochastic demand, the total weighted revenue in all measurement duration under placement P can be written as:

$$R(P) = w_{sat} \sum_{i=1}^{I} min\left(L, d^i\right) + w_{loc} \sum_{i=1}^{I} \sum_{j=1}^{J} min\left(L_j, d_j^i\right). \quad (3)$$

Then for stochastic demand, the revenue can be written as the expected value of all possible revenue:

$$E(R(P)) = w_{sat} \sum_{i=1}^{I} E\left(min\left(L, d^i\right)\right) + w_{loc} \sum_{i=1}^{I} \sum_{j=1}^{J} E\left(min\left(L_j, d_j^i\right)\right). \quad (4)$$

Then given a global budget C, the objective of algorithm is to find the optimal placement P that maximizes the expected revenue:

$$\max\left(E(R(P))\right) \qquad s.t. \quad \sum_{j=1}^{J} f_j^c(L_j) = C. \quad (5)$$

IV. Algorithm

Due to the existence of non-linear target and constraint functions, the problem described in (5) is a high complicated non-linear optimization one. And the value of J can range from 10 (number of data centers in cloud platform) to more than 100 (number of nodes in content delivery network), with each j contain numbers of measurement duration and different charging function $f_j^c(\cdot)$. Furthermore, distributions can be correlated, which make the problem more complex. We need to find properties of solutions to devise efficient algorithms. First we introduce following lemma to get transformed problems:

Lemma 4.1 For any continuous non-negative random variable X with CDF $cdf_X(\cdot)$ and constant C, it has: $E(\min(C, X)) = \int_0^C (1 - cdf_X(v))dv$.

Proof of Lemma 4.1: Note that for given constant C, it has:

$$Pr(\min(C, X) \le v) = \begin{cases} Pr(X \le v) & v \le C \\ 1 & v > C \end{cases}. \quad (6)$$

Denote random variable $Y = \min(C, X)$, and X and Y 's cumulative distribution function (CDF) as $cdf_X(\cdot)$ and $cdf_Y(\cdot)$, then (6) can be rewritten as:

$$cdf_Y(v) = \begin{cases} cdf_X(v) & v \le C \\ 1 & v > C \end{cases}. \quad (7)$$

According to [21], for every real-valued non-negative random variable Z with probability density function $cdf_Z(\cdot)$, it has $E(Z) = \int_0^\infty (1 - cdf_Z(v))dv$. Substituting Z by Y, we can get $E(\min(C, X)) = E(Y) = \int_0^\infty (1 - cdf_Y(v))dv = \int_0^C (1 - cdf_X(v))dv$. Then the lemma is proved.

Based on lemma 4.1, the revenue in (4) can be rewritten as:

$$\begin{aligned} E(R(P)) = {} & w_{sat} \sum_{i=1}^{I} \int_0^L (1 - cdf^i(n))dn) + \\ & w_{loc} \sum_{i=1}^{I} \sum_{j=1}^{J} \int_0^{L_j} (1 - cdf_j^i(n))dn \end{aligned}. \quad (8)$$

A. Subproblem 1

Although the original problem stated in (5) (denoted as SP_0) has no global optimal condition, when we investigated a similar and simper problem (denoted as SP_1):

$$\max\left(E(R(P))\right) \qquad s.t. \quad \sum_{j=1}^{J} L_j = U. \quad (9)$$

We found problem SP_1 has a optimal condition which can be solved efficiently, and the solution of problem SP_1 provides a lower bound for maximal revenue in solution to the original problem SP_0, as stated in lemma 4.2:

Lemma 4.2 Denote one optimal solution to SP_1 as $P^U = \{L_j^U\}$ with maximal revenue $E(R(P^U))$. If $\sum_{j=1}^{J} f_j^c(L_j^U) = C$, set the optimal solution to the original problem as $P = \{L_j^C\}$ with maximal revenue $E(R(P^C))$, it has: (1) $E(R(P^C)) \geq E(R(P^U))$ and (2) $\sum_{j=1}^{J}(L_j^C) >= U$.

Proof of Lemma 4.2: For (1), since the optimal solution of problem SP_1 is a feasible solution of original problem, then the revenue of optimal solution of original problem is at least larger than that of problem SP_1, then (1) is proved. For (2), we can easily deduce that for optimal solution to two problem SP_1 with different U, it has $E(R(P^{U_1})) \leq E(R(P^{U_2}))$ when $U_1 \leq U_2$. Therefore, if we assume the optimal solution of original problem has $\sum_{j=1}^{J}(L_j^C) = U' < U$, then it has $E(R(P^C)) = E(R(P^{U'})) < E(R(P^U))$, which gets a contradiction that the optimal solution is not better than a feasible solution. Then lemma 4.2 is proved.

Lemma 4.2 means that by solving problem SP_1, we can quickly approach optimal solution to the original problem SP_0, and then fine tune the placement to get solution closer to optimal ones. We then need to solve the problem SP_1:

Lemma 4.3 In the optimal condition of problem SP_1, all partial derivative functions equal: $cdf_1^U(L_1) = cdf_2^U(L_2) = ... = cdf_J^U(L_J) = v$, where $v = \sum_{i=1}^{I}(1 - cdf_j^i(L_j))$ is the optimal variable, and $cdf_j^U(L_j) = \sum_{i=1}^{I} cdf_j^i(L_j)$.

Proof of Lemma 4.3: define $f_j(L_j)$ as $f_j(L_j) = \int_0^{L_j}(1 - cdf_j^i(n))dn$, with its derived function is $f_j'(L_j) = 1 - cdf_j^i(L_j) = cdf_j^U(L_j)$. Then the solution to problem SP_1 in (8) is equal to that of the following problem: $\max(\sum_{j=1}^{J} f_j(L_j)) \quad s.t. \sum_{j=1}^{J} L_j = U$. The problem is in the form of a traditional non-linear programming problem, where the optimal condition is all derived function equals, and then lemma 4.3 is proved.

Based on lemma 4.3, we use binary search to narrow the scope and find the solution close enough to the global budget C. The algorithm is outlined in Figure 1, where in each iteration the cost (c) is first calculated using the given optimal

Input: demand distribution $(cdf_j^U)^{-1}(\cdot)$.

Output: amount of resources placed in each region L_j, lower bound of total resource amount U^C.

```
scope = I / 2.
v = 0.
while (true)
{
  v = v + scope.
  for ( j = 1 to J )
  {
    L_j = (cdf_j^U)^{-1}(v).
  }
  scope = scope / 2.
  c = Σ_{j=1}^{J} f_j^c(L_j).
  if ( c is close enough to C )
  {
    U^C = Σ_{j=1}^{J} L_j.
    return U^C and all L_j.
  }
  else if( c < C ) {
    v = v − scope.
  }
  else {
    v = v + scope.
  }
}
```

Fig. 1. The algorithm to solve subproblem 1.

variable v (line 4-9), then c is compared with target cost C to narrow search scope or terminate algorithm (line 10-17).

In implementation of algorithm 1 in Figure 1, all input functions can be recorded in hash map data structure, and its time complexity of querying operation is $O(1)$. Although $(cdf_j^U)^{-1}(\cdot)$ may give a value range instead of single value, we

Input: demand distribution $(cdf_j^U)^{-1}(\cdot)$, outer iteration number K_1, inner iteration number K_2.

Output: amount of resources placed in each region L_j.

Get initial placement $L_j(1 \le j \le J)$ using algorithm 1.

for($k_1 = 1$ to K_1){

$exch\_cost = k_1 C / (K_1 J)$.

for($k_2 = 1$ to K_2){

for($j = 1$ to J){

record added revenue to $list\_add[j]$ if increase the budget of region j by $exch\_cost$.

record reduced revenue to $list\_red[j]$ if reduce the budget of region j by $exch\_cost$.

}

sort $list\_add$ in ascending order, find the first region j_1 with added revenue rev_add.

sort $list\_red$ in descending order, find the first region j_2 with reduced revenue rev_red.

\If{($rev\_add > rev\_red$)}{

apply the exchange and update L_{j_1} and L_{j_2}.

}

}

}

Fig. 2. The final resource scheduling algorithm.

can choose the lowest value to find the highest lower bound U^C.

B. Solution to original problem

We then investigate problem of how to approach optimal solution of SP_0 from the solution of SP_1. But as problem SP_0 has no global optimal condition, exhausting every local optimal solution is of high time complexity. We choose to fine tune the solution of SP_1 by exchanging resource between different regions. The final algorithm is given in Figure. 2, which is called HCRS (Heterogeneous Cost models oriented cloud Resource Scheduling algorithm). Where in inner iteration (line 4-14), we first calculate the added and reduced revenue of changing L_j by $exch\_cost$ (line 5-8), then pair and update the region with largest adding revenue (j_1) and smallest reducing revenue (j_2) (line 9-13). The out iteration just increases $exch\_cost$ and tries inner iteration again.

V. Experiments

To investigate the effectiveness of HCRS, we evaluate HCRS algorithm to compare the effectiveness of HCRS with traditional mean-based algorithm. Similar to the experiment settings of stochastic demand oriented works [4], we consider demands that follow *Zipf* distribution, i.e., the probability of a single request dispatched to region j is:

$$p_j = \frac{j^{-e}}{\sum_{n=1}^{J} n^{-e}}, \tag{10}$$

where $e > 0$ is a real number.

Without loss of generality, we initialize the expectation of D_j^i to be randomized using Poisson distribution with a rate of $p_j \lambda$, where λ is the expected number of total requests. To find the effectiveness of algorithm with low complexity, we set low cycle count of $K_1 = 10$ and $K_2 = 100$.

Firstly we compare the effect of mean-based method with HCRS under different budgets. In mean-based method, the number of resource in region j is proportional to the mean demand, i.e., to solve the following problem:

$$\sum_{j=1}^{J} f_j^c(rE(D_{j'}^{i'})) = C, \tag{11}$$

where r is the same ratio of resource to demand in each region.

The result is shown in Figure 3. Here we use the cost to fulfill mean demand of each region as C, the x-axis means the budgets represented using proportion of C, and the y-axis means the percent of increased revenue using HCRS upon mean-based method. The *Zipf* parameter is set to 1.0. It is shown that HCRS can always outperform mean-based methods under different budgets. Furthermore, it is observed that, along with the increasing of budget, revenue increasement decreases significantly. The reason is that the number of unsatisfied demands decrease with more resources, which leads

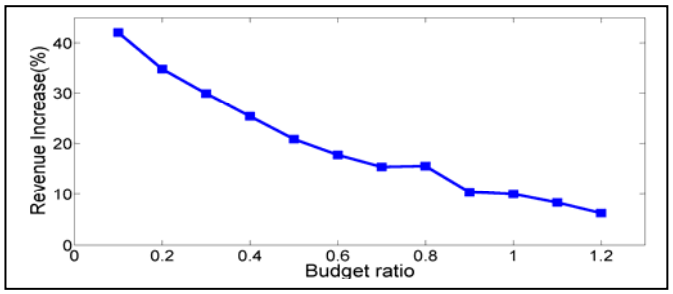

Fig. 3. Revenue increase from mean-based method under different budgets.

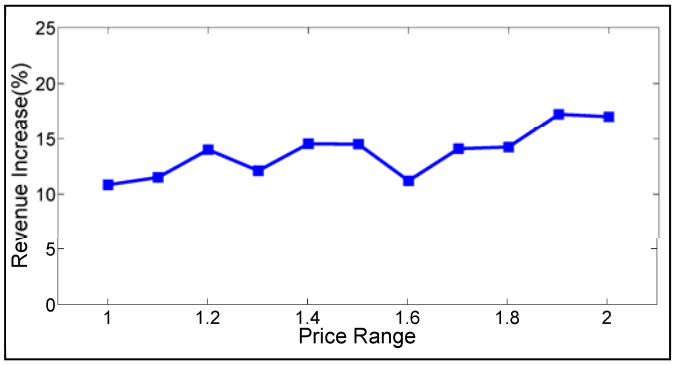

Fig. 4. Revenue increase from mean-based method under different price range.

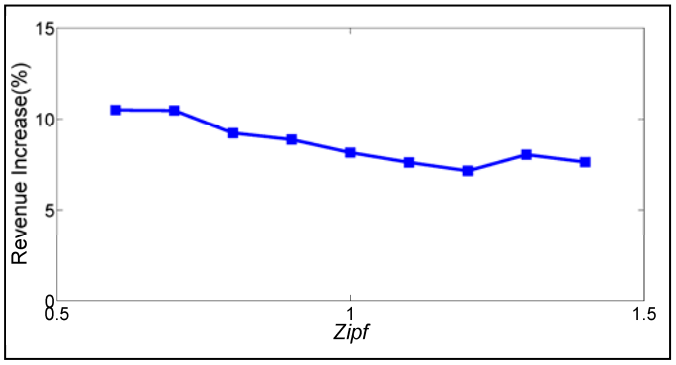

Fig. 5. Revenue increase from mean-based method under different *Zipf* parameter.

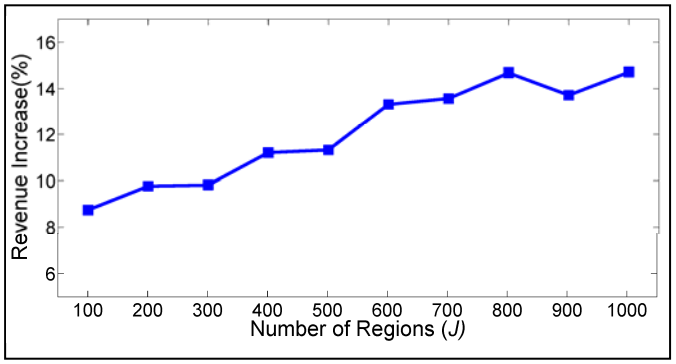

Fig. 6. Revenue increase from mean-based method under different *J*.

to the decreasing of room for optimization. Since resources cannot be surplus enough in most of real scenarios, HCRS can still be used to gain more revenue in common scenarios.

Secondly we explore to find out how the different pricing model in each region affects the revenue. According to the pricing model in Amazon EC2, where the price of most expensive instance can be nearly two times of that of cheapest instance, thus we choose linear pricing model with unit price ranges from 1.0-2.0. We also use the cost to fulfill mean demand of each region as C, the x-axis means the range of unit price among all regions, and the y-axis means the percent of increased revenue using HCRS upon mean-based method. The *Zipf* parameter is set to 1.0, and budget is set to $0.5C$. The result is shown in Figure 4. It can be observed that, along with the increasing of price range, revenue increased slowly, which perhaps due to the small price range here. Since in realistic scenario the price range cannot be very large, that means the impact of price difference is small.

Then we investigate how *Zipf* parameter affects the revenue. We consider a wide range of *Zipf* with varied values from 0.6 to 1.4.The result is shown in Figure 5. We can observe that along with the increase of *Zipf* parameter, the revenue decreased accordingly. Note that higher *Zipf* means greater popularity difference between different j. Since improving result needs more computation, the result indicates that it needs more computation to elaborate result for demands with high *Zipf*.

The number of regions varies a lot in realistic systems, ranging from 100 to 1000, which will affect the result of HCRS. We test and calculate the revenue increase under different J. The result is shown in Figure 6. We can observe an obvious improvement of revenue with greater J, which dues to that greater J provides more opportunities for revenue increasing in cycles of HCRS algorithm.

Above all, it is shown in experiment that:

(1) HCRS is more effective when resource is scarce (i.e., fewer budgets), which means HCRS is especially effective for situation of resource overload. In another word, HCRS can help improve performance without adding more budget in case of request burst.

(2) The effectiveness of HCRS is slightly affected by unit price. That means HCRS can adapt to the common price range in different cloud platform, thus can be deployed across cloud platforms.

(3) The *Zipf* parameter has little impact on the effectiveness of HCRS, which indicates that HCRS can accommodate different demand distribution across regions.

(4) The number of regions can affect HCRS. It is shown that the result of HCRS is improved with more regions, i.e., the revenue increase can compensate the increased complexity when number of regions increases.

VI. Conclusion

We propose an efficient stochastic demand oriented cost optimization resource placement algorithm (called HCRS). Our analysis and the preliminary experiments indicate that with lower computation complexity, HCRS can increase revenue by up to 42% when compared with mean demand based algorithm. HCRS can alleviate the situation of resource overload without increasing budget, and is expected to accommodate demands with different demand distribution across regions.

Our future works may include: more effective resource adjust algorithm to improve the algorithm result and efficiency; more sophisticated scenarios with multiple types of resources.

ACKNOWLEDGMENT

This paper is supported by the National Natural and Science Foundation of China (61472460, U1504607), Natural Science Research of the Education Department of Henan Province (17A520004, 16A520006), Program for Innovative Research Team (in Science and Technology) in University of Henan Province (17IRTSTHN011), Plan of Nature Science Fundamental Research in Henan University of Technology (2014JCYJ04, 2015XTCX03), Key Laboratory of Grain Information Processing and Control (Henan University of Technology)， Ministry of Education (KFJJ-2016-104),Doctor Foundation of Henan University of Technology (No.2012BS011).

REFERENCES

[1] Netflix home page (http://www.netflix.com).

[2] Facebook home page (http://www.facebook.com).

[3] YouTube home page (http://www.youtube.com).

[4] Y. Rochman, H. Levy, and E. Brosh, “Resource placement and assignment in distributed network topologies,” Proceedings of IEEE INFOCOM, 2013.

[5] Y. Rochman, H. Levy, and E. Brosh, “Efficient resource placement in cloud computing and network applications,” to be published in INFOCOM, 2014 Proceedings IEEE, 2014.

[6] W. Wei, Y. Zhang, Y. Liu, and Z. Qin, “Frp: a fast resource placement algorithm in distributed cloud computing platform,” Concurrency and Computation: Practice and Experience, vol.28, no.5, pp.1399–1416, 2016. cpe.3654.

[7] Amazon EC2 home page (http://aws.amazon.com/ec2).

[8] Microsoft Azure home page (http://www.windowsazure.com).

[9] F. Wang, J. Liu, and M. Chen, “Calms: Cloud-assisted live media streaming for globalized demands with time/region diversities,” INFOCOM, 2012 Proceedings IEEE, pp.199–207, 2012.

[10] [X. Nan, Y. He, and L. Guan, “Queueing model based resource optimization for multimedia cloud,” Journal of Visual Communication and Image Representation, vol.25, no.5, pp.928–942, 2014.

[11] Y. Zhao, H. Jiang, K. Zhou, Z. Huang, and P. Huang, “Meeting service level agreement cost-effectively for video-on-demand applications in the cloud,” INFOCOM, 2014 Proceedings IEEE, pp.298–306, IEEE, 2014.

[12] W. Wang, D. Niu, B. Liang, and B. Li, “Dynamic cloud instance acquisition via iaas cloud brokerage,” IEEE Transactions on Parallel and Distributed Systems, vol.26, no.6, pp.1580–1593, 2015.

[13] D.M. Divakaran and M. Gurusamy, “Towards flexible guarantees in clouds: Adaptive bandwidth allocation and pricing,” IEEE Transactions on Parallel and Distributed Systems, vol.26, no.6, pp.1754–1764, 2015.

[14] A. Alasaad, K. Shafiee, H.M. Behairy, and V.C. Leung, “Innovative schemes for resource allocation in the cloud for media streaming applications,” IEEE Transactions on Parallel and Distributed Systems, vol.26, no.4, pp.1021–1033, 2015.

[15] A. Beloglazov and R. Buyya, “Managing overloaded hosts for dynamic consolidation of virtual machines in cloud data centers under quality of service constraints,” IEEE Transactions on Parallel and Distributed Systems, vol.24, no.7, pp.1366–1379, 2013.

[16] L. He, D. Zou, Z. Zhang, C. Chen, H. Jin, and S.A. Jarvis, “Developing resource consolidation frameworks for moldable virtual machines in clouds,” Future Generation Computer Systems, vol.32, pp.69–81, 2014.

[17] D. Niu, H. Xu, B. Li, and S. Zhao, “Quality-assured cloud bandwidth auto-scaling for video-on-demand applications,” Proceedings of IEEE INFOCOM, pp.460–468, IEEE, 2012.

[18] B. Urgaonkar, P. Shenoy, and T. Roscoe, “Resource overbooking and application profiling in shared hosting platforms,” ACM SIGOPS Operating Systems Review, vol.36, no.SI, pp.239–254, 2002.

[19] W. Wei, L. Yang, and Y. Zhang, “Trlms: two-stage resource scheduling algorithm for cloud based live media streaming system,” IEICE TRANSACTIONS on Information and Systems, vol.97, no.7, pp.1731–1734, 2014.

[20] W. Wei, Y. Zhang, Y. Liu, and Z. Qin, “Frp: a fast resource placement algorithm in distributed cloud computing platform,” Concurrency and Computation: Practice and Experience, 2015.

[21] S.M. Ross, Introduction to probability models, Academic press, 2006.