

International Journal of Computer Science and Mobile Computing

A Monthly Journal of Computer Science and Information Technology

ISSN 2320-088X

IJCSMC, Vol. 3, Issue. 7, July 2014, pg.189 – 194

RESEARCH ARTICLE



ERROR-TOLERANT RESOURCE ALLOCATION AND PAYMENT MINIMIZATION FOR CLOUD SYSTEM

Suneetha Kurapati

Student

M.TECH, Department of CSE,

Maheshwara Institute of Technology,

Suneetha50779@gmail.com

P. Prem Kumar

Assistant Professor

Department of CSE

Maheshwara Institute of Technology,

prem1218@gmail.com

Abstract: Virtual machine (VM) technology being greater and fully developed, compute resources in cloud systems can be partitioned in fine granularity and allocated on demand, which contributes three technologies such as, Formulating a deadline-driven resource allocation problem based on the cloud environment facilitated with VM resource isolation technology, and also to minimize users' payment. Analyzing the upper bound of task execution length based on the possibly inaccurate workload prediction, it further proposed an error-tolerant method to guarantee task's completion within its deadline and Validating its effectiveness over a real VM-facilitated cluster environment under different levels of competition.

Keywords: VM multiplexing, Resource allocation, Error tolerance, Payment minimization

I. INTRODUCTION

Cloud computing is a model for enabling convenient, on-demand network access to a shared pool of configurable computing resources (software, data storage, network, etc.) that can be rapidly provisioned and released with minimal management effort or service provider interaction. Cloud computing [1], is a comprehensive solution that delivers IT as a service. It is an Internet-based computing solution where shared resources are provided like electricity distributed on the electrical grid. Computers in the cloud are configured to work together and the various applications use the collective computing power as if they are running on a single system. The characteristics of cloud computing are: user friendliness, virtualization, Internet centric, variety of resources, automatic adaptation, scalability, resource optimization, pay-per-use, service SLAs (Service-Level Agreements)[15] and infrastructure SLAs.

Reduced cost: Cloud computing can reduce both capital expense (CapEx) and operating expense (OpEx) costs because resources are only acquired when needed and are only paid for when used.

Refined usage of personnel: Using cloud computing frees valuable personnel allowing them to focus on delivering value rather than maintaining hardware and software.

Robust scalability: Cloud computing allows for immediate scaling, either up or down, at any time without long-term commitment. The resource allocation in cloud computing is much more complex than in other distributed systems like Grid computing platform. In a Grid system [3], it is improper to share the compute resources among the multiple applications simultaneously running atop it due to the inevitable mutual performance interference among them. Whereas, cloud systems usually do not provision physical hosts directly to users, but leverage virtual resources isolated by VM technology [4], [5], [6]. Not only can such an elastic resource usage way adapt to user’s specific demand, but it can also maximize resource utilization in fine granularity and isolate the abnormal environments for safety purpose. Some successful platforms or cloud management tools leveraging VM resource isolation technology include Amazon EC2 [7] and OpenNebula .

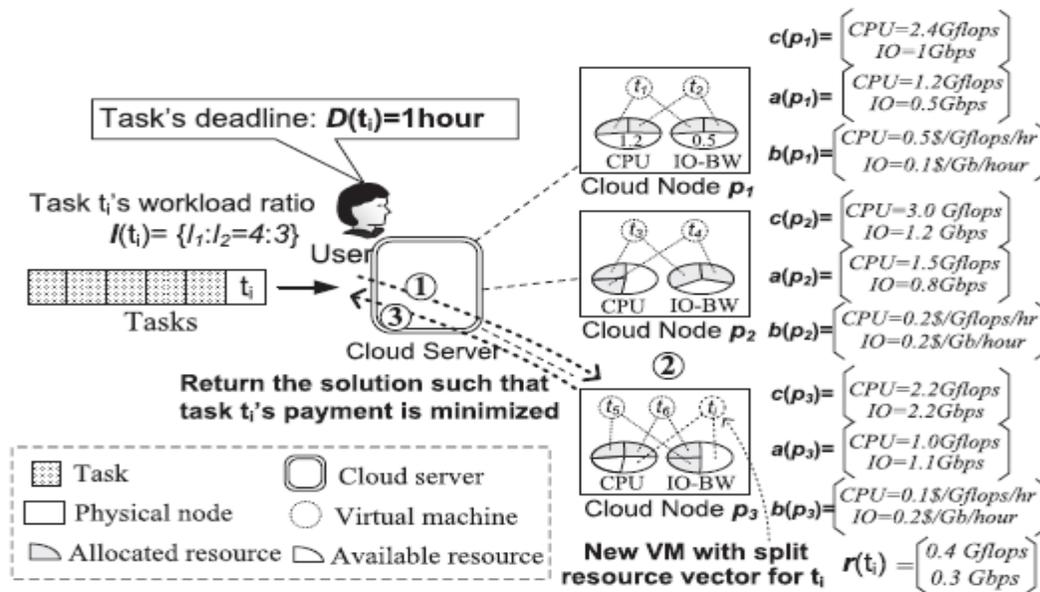


Fig 1.0 Resource allocation in cloud system

II. PROPOSED SYSTEM

This scheme proposed a novel resource allocation algorithm for cloud system that supports VM-multiplexing technology, aiming to minimize user’s payment of task within its deadline. It can prove that the output of the algorithm is optimal based on the KKT condition[2], which means any other solutions would definitely cause larger payment cost. In addition, this paper analyze the approximation ratio for the expanded execution time generated by the algorithm to the user-expected deadline, under the possibly inaccurate task property prediction. When the resources provisioned are relatively sufficient, It can guarantee task’s execution time[11] always within its deadline even under the wrong prediction about task’s workload characteristic.

- 1) It formulate a deadline-driven resource allocation problem based on the cloud environment facilitated with VM resource isolation technology, and also proposed a novel solution with polynomial time, which could minimize users' payment in terms of their expected deadlines.
- 2) By analyzing the upper bound of task execution length based on the possibly inaccurate workload prediction, it further proposed an error-tolerant method to guarantee task's completion within its deadline.
- 3) It validates its effectiveness over a real VM-facilitated cluster environment under different levels of competition.

III. OPTIMALITY ANALYSIS WITH INACCURATE INFORMATION

In this section, we focus on such a question: what is the final upper bound of task execution length as compared to its predefined deadline D , when running it using the resource vector allocated under Algorithm 1 with inaccurately predicted workload information?

A Problem Description

Although Algorithm 1's output is proved optimal, such a result relies on a strong condition, i.e., accurate task's workload vector. That is, each user needs to precisely predict the execution property (i.e., workload ratio) for his/ her task, before constructing the resource allocation with minimized payment for its execution under a user-specified deadline. In some cases, the execution property could be easily estimated accurately. For instance, we can decide the workload ratio between the data to be read/written from/ to disk and those to be downloaded/uploaded via network by comparing their data sizes. In many other cases, however, the execution property cannot be accurately estimated, such as computation-intensive applications whose execution times highly depend on the CPU cycles to consume.

IV. SYSTEM IMPLEMENTATION

A Optimal Resource Allocation

Based on the convex optimization theory[2],[3], the Lagrangian function of the problem could be formulated as (1), where θ and μ_k are corresponding Lagrangian multipliers. Note that θ is a constant defined in (1) and r is the abbreviation of $r(t_i)$ as stated above

$$F1(R) = \frac{1}{R} \left(\sum_{k=1}^R b_k r_k \right) \left(\theta \sum_{k=1}^R \frac{l_k}{r_k} \right) + \lambda \left(\theta \sum_{k=1}^R \frac{l_k}{r_k} - D \right) + \sum_{k=1}^R \mu_k (r_k - a_k)$$

(1) Where,

R =Execution Dimension,

B_k =Price Vector,

R_k =Resource Vector,

L_k =Workload Vector,

D =Deadline,

A_k =Available Vector

According to Karush-Kuhn-Tucker conditions (i.e., the necessary and sufficient condition of the optimization).

$$\lambda \geq 0, \mu_K \geq 0, K = 1, 2 \dots R$$

$$\sum_{I=1}^R \theta \frac{l_I}{r_I} \leq D$$

$$\lambda \left(\sum_{I=1}^R \theta \frac{l_I}{r_I} - D \right) = 0$$

$$r_K \leq a_K(P_S), K = 1, 2 \dots R; S = 1, 2, \dots n$$

$$\mu_K (r_K - a_K(P_S)) = 0, K = 1, 2 \dots R; S = 1, 2 \dots n$$

$$\frac{dF1}{dr_k} = \frac{1}{R} \left(\left(\sum_{i=1}^R b_i r_i \right) \cdot \frac{-l_k}{r_k^2} + b_k \sum_{i=1}^R \frac{l_i}{r_i} + -\lambda \frac{l_k}{r_k^2} + \mu_k \right) = 0$$

K=1,2...R

Where,

R=Resource,

Bk=Price Vector,

Rk=Resource Vector,

Lk=Workload Vector,

D=Deadline,

Ak=Available Vector

Optimal allocation algorithm

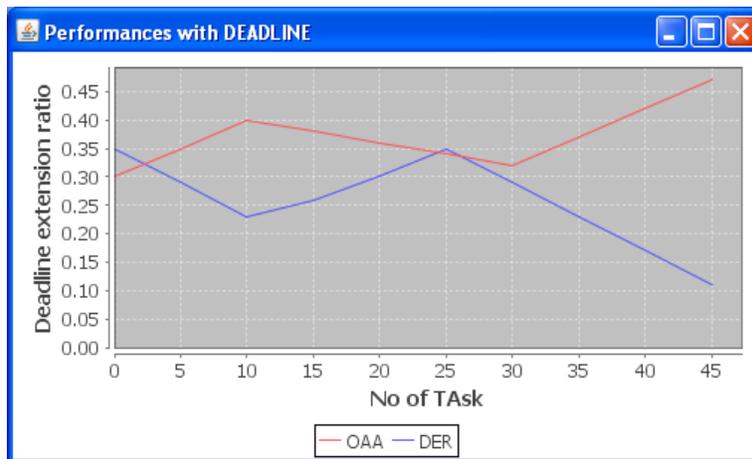
- Input: D(ti); Output: execution node ps, r*(ti)
- $\Gamma = \Pi, C = D(ti), r^* = \phi$ (empty set);
- Repeat
- $r\Gamma^*(ti, ps) = \text{CO-STEP}(\Gamma, c)$;
- on Γ^*
- $\Omega = \{ dk / dk \in \Gamma \ \& \ r \ k(*) (ti, ps) > ak(ps) \}$;
- $\Gamma = \Gamma \setminus \Omega / * \Gamma$ take away $\Omega^* /$
- $C = C - \theta l k a k d k \in \Omega / * \text{Update } C^* /$
- $r^*(ti, ps) = r^*(ti, ps) \cup \{ r \ k(*) = ak(ps) | dk \in \Omega \ \& \ ak(ps) \}$
- is dk_s upper bound};
- until ($\Omega = \phi$);
- $r^*(ti, ps) = r^*(ti, ps) \cup r\Gamma^*(ti, ps)$
- end for

- Select the smallest $p(t_i)$ by traversing the candidate solution set;
- Output the selected node p_s and resource allocation $r^*(t_i, p_s)$;

Based on the Algorithm 1, it is obvious that the local optimal resource allocation for t_i to be executed on specified node p_s is the most crucial part. In fact, the final outputted resource allocation solution of the whole algorithm will be globally optimal around the whole system as long as each local process on a specified node (line 2-10) can be proved as optimal resource allocation. Consequently, this will intensively discuss the local divisible-resource allocation by specifying a particular execution node, in the following text. Although Algorithm 1 is proved optimal for minimizing the payment cost within user-defined deadline of task still may not be guaranteed due to two factors, either bounded available resources or inaccurate workload vector information about the task. IT proposed the following lemma, which provides a necessary and sufficient condition of guaranteeing [8] the task’s deadline given accurate prediction and relatively sufficient resources.

V. RESULTS AND DISCUSSION

A RESULTS: OAA and DER



CONCLUSION AND FUTURE WORK

A CONCLUSION:

This paper proposes a novel resource allocation algorithm for cloud system that supports VM-multiplexing technology, aiming to minimize [6] user’s payment of task and also endeavor to guarantee its execution deadline meanwhile. This is proven that the output of this algorithm is optimal based on the KKT condition, which means any other solutions would definitely cause larger payment cost.

B FUTURE ENHANCEMENT:

In the future, the plan to integrate this algorithm with stricter/original deadlines into some excellent management tools, for maximizing the system-wide performance. To improve the resource utilization and reduce the user payment the future work to fix the KKT based on dual parameters like deadline, cost or bandwidth.

REFERENCES

- [1]. Armbrust.M, Fox.A, Griffith.R, Joseph.A.D, Katz.R.H, Konwinski.A, Lee.G, Patterson.D.A, Rabkin.A, Stoica.I, and Zaharia.M,(2009), —Above the Clouds: A Berkeley View of Cloud Computing, Technical Report UCB/EECS-2009-28, EECS Dept., Univ. California, Berkeley.
- [2]. Boyd.S and Vandenberghe.L,(2009), —Convex Optimization, Cambridge Univ. Press.
- [3]. Chang.F, Ren.J, and Viswanathan.R, (2010). —Optimal Resource Allocation in Clouds, Proc. IEEE Int'l Conf. Cloud Computing, pp. 418-425.
- [4]. Gupta.D, Cherkasova.L, Gardner.R, and Vahdat.A, (2006), —Enforcing Performance Isolation across Virtual Machines in Xen, Proc. bACM/IFIP/USENIX Int'l Conf. Middleware (Middleware '06), pp. 342-362.
- [5]. Huang.L, Jia.J, Yu.B, Chun.B.G, Maniatis.P, and Naik.M,(2010), —Predicting Execution Time of Computer Programs Using SparsePolynomial Regression, Proc. 24th Conf. Neural Information Processing Systems (NIPS '10), pp. 1-9.
- [6]. Mao.M and Humphrey.M,(2011), —Auto-Scaling to Minimize Cost and Meet Application Deadlines in Cloud Workflows, Proc. Int'l Conf. High Performance Computing, Networking, Storage & Analysis (SC '11), pp. 49:1-49:12.
- [7]. Matthews.J.N, Hu.W, Hapuarachchi.M, Deshane.T, Dimatos.D, Hamilton.G, McCabe.M, and Owens.J, (2007), —Quantifying the Performance Isolation Properties of Virtualization Systems, Proc. Workshop Experimental Computer Science (ExpCS '07).
- [8]. McElvany.M.C and Stotts.P.D, (1991), —Guaranteed Task Deadlines for Fault-Tolerant Workloads with Conditional Branches, Real-Time Systems, vol. 3, no. 3, pp. 275-30.
- [9]. Meng.X, Isci.C, Kephart.J, Zhang.L, Bouillet.E, and Pendarakis.D, (2010), —Efficient Resource Provisioning in Compute Clouds via VM Multiplexing, Proc. Seventh Int'l Conf. Autonomic Computing (ICAC '10), pp. 11-20.
- [10]. Nathuji.R, Kansal.A, and Ghaffarkhah.A,(2010), —Q-Clouds : Managing Performance Interference Effects for Qos-Aware Clouds, Proc. European Conf. Computer Systems (EuroSys '10), pp. 237-250.