Optimal Provisioning of Resource in a Cloud Service

Yee Ming Chen[1] Shin-Ying Tsai

Department of Industrial Engineering and Management, Yuan Ze University
135 Yuan-Tung Rd., Chung-Li, Tao-Yuan, Taiwan, ROC.

Abstract
Cloud service allows enterprise class and individual users to acquire computing resources from large scale data centers of service providers. This cloud service is more involved in purchasing and consuming manners between providers and users than others. However, Cloud service providers charge users for these services. Specifically, to access data from their globally distributed storage edge servers, providers charge users depending on the user's location and the amount of data transferred. User applications may incur large data retrieval and execution costs. Therefore, optimizing execution time, the cost arising from data transfers between resources as well as execution costs should be taken into account. In this paper, we present a discrete Particle Swarm Optimization (DPSO) approach for tasks allocation. We construct application Amazon EC2 as an example and simulation with Cloud based compute and transmission resources. Experimental studies illustrate that the proposed method is more efficient and surpasses those of mathematical programming and reflecting the actual benefit of saving with the total cost as well as tasks allocation.
Keywords: *Particle Swarm Optimization, Resource Allocation, Cloud service provider.*

1. Introduction

Cloud computing is a modality of computing characterized by on demand availability of resources in a dynamic and scalable fashion. The term resource here could be used to represent infrastructure, platforms, software, services, or storage. Cloud computing services allow users to lease computing resources from large scale data centers operated by service providers. Using cloud services, cloud users can deploy a wide variety of applications dynamically and on-demand. Most cloud service providers use machine virtualization to provide flexible and cost effective resource sharing. The cloud service provider is responsible to make the needed resources available on demand to the cloud users. It is the responsibility of the cloud service provider to manage its resources in an efficient way so that the cloud user needs can be met when needed at the desired Quality of Service (QoS) level[1]. Recently, many companies, such as Amazon, Google and Microsoft, have launched their cloud service businesses. Most cloud service providers use machine virtualization techniques to provide flexible and cost-effective resource sharing among users. Virtual machine(VM)instances normally share physical processors and I/O interfaces with other instances. It is expected that virtualization can impact the computation and communication performance of cloud services. Although most commercial providers present VM performance criteria to customers, it is difficult for management systems to assure VMs of their minimize execution cost or maximum assigned resources. If the tasks of VMs, for example, suddenly change from idle to active, the locations of VMs cannot be optimized again to meet the change[2]. In this paper, we propose meta-heuristic optimization approach based on Particle Swarm Optimization (PSO) for finding the near optimal tasks allocation with reasonable time. The approach is to dynamically generate an optimal task allocation so as to complete the tasks in a minimum period of time as well as utilizing the resources in an efficient way. The rest of the paper is organized as follows. Section 2 deals with some theoretical foundations related to tasks allocation model. In Section 3, we describe the proposed DPSO based algorithm in detail. Experimental results are presented in Section 4 and some conclusions and future works are provided towards the end.

2. Provisioning of Resources in a Cloud Environment

Cloud computing services are often roughly classified into a hierarchy of as a service terms as following[3]:

Infrastructure as a Service (IaaS) is providing general on-demand computing resources such as virtualized servers or various forms of storage (block, key/value, database, etc.) as metered resources. This can often be seen as a direct evolution of shared hosting with added on-demand scaling via resource virtualization and use-based billing.

Platform as a Serv ice (P aaS) is providing an existent managed higher-level software infrastructure for building particular classes of applications and services. The platform includes the use of underlying computing resources, typically billed similar to IaaS products, although the infrastructure is abstracted away below the platform.

Software as a Service (SaaS) is providing specific, already-created applications as fully or partially remote services. Sometimes it is in the form of web-based applications and other times it consists of standard non-remote applications with Internet-based storage or other network interactions.

EC2 and other server clouds follow an IaaS model, in which the cloud users rent virtual servers and selects or controls the software for each virtual server tasks[4]. Every cloud service providers might have a unique way of managing and tasks allocation must ensure that they do not conflict with the resource owner's policies. In the worst-case situation, the cloud service providers might charge different prices to different cloud users for their resource usage and this might vary from time to time. Mathematical programming approaches [5] using column generation or branch-and-bound techniques can solve the tasks allocation problem[6]. However, the general n-processor tasks allocation has been found to be NP-complete[7]. Therefore, finding exact optimum solutions to large-scaled tasks allocation problem is computationally prohibitive. The development of meta-heuristic optimization theory has been flourishing during the last decade. Particularly, with its sound exploration ability of both global and local optimal solutions, some new search techniques involving nature-inspired meta-heuristics have become the new focus of resource allocation research. As mentioned in [8] scheduling is NP-complete. Meta-heuristic methods have been used to solve well-known NP-complete problems. Efficient Meta-heuristic methods, which are used frequently, are simulated annealing (SA) [9], genetic algorithm (GA) [10], ant colony optimization (ACO) [11] and particle swarm optimization (PSO)[12].

In this study, we consider the tasks allocation with the following scenarios(figure 1). The processors in the system are heterogeneous and they are capacitated with various units of memory and processing resources. Hence, a task will incur different execution cost if it is executed on different processors. On the other hand, all of the communication links are assumed to be identical and some communication cost between two tasks will be incurred if there is a communication need between them and they are executed on different processors.

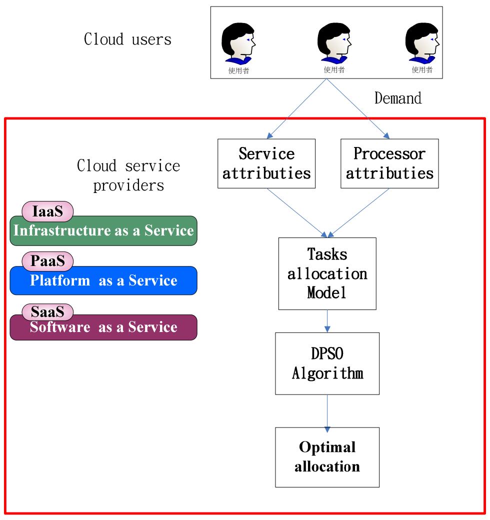

Figure 1 The framework of tasks allocation process

In this paper, a version of discrete particle swarm optimization (DPSO) is proposed for cloud service provider's tasks allocation and the goal of allocation is to minimize the execution cost and communication cost mentioned above simultaneously.

2.1 Tasks allocation Model

The Tasks allocation model [13,14]is an integer program with a quadratic objective function (1) which represents the total execution cost and communication cost, respectively.

$$Min\ C(X) = \sum_{i=1}^{t}\sum_{k=1}^{p} ec_{ik}x_{ik} + \sum_{i=1}^{t-1}\sum_{j=i+1}^{t} cc_{ij}(1-\sum_{k=1}^{p} x_{ik}x_{jk}) \qquad (1)$$

Constraints：

$$\sum_{k=1}^{n} x_{ik} = 1, \qquad \forall_i = 1,2,\cdots,t \qquad (2)$$

$$\sum_{i=1}^{t} r_i x_{ik} \le R_k, \quad \forall_k = 1,2,\cdots,p \qquad (3)$$

$$\sum_{i=1}^{t} m_i x_{ik} \le M_k, \forall_k = 1,2,\cdots,p \qquad (4)$$

$$x_{ik} \in (0,1) \qquad (5)$$

Constraint (2) states that each task should be allocated to exactly one processor. Constraints (3) and (4) ensure that processing resource and the memory capacity of each processor is no less than the total amount of resource demands of all of its allocated tasks. The last constraint (5) guarantees that x_{ik} are binary decision variables. As mentioned in the previous section the goal of the tasks allocation is to minimize the total execution cost and communication cost simultaneously.

3. Proposed Discrete Particle Swarm Optimization Algorithm

In this section we propose a version of discrete particle swarm optimization for tasks allocation. Particle needs to be designed to present a sequence of tasks in available cloud service providers. Also the velocity has to be redefined. Details are given what follows.
In our method solutions are encoded in a $t \times p$ matrix, called position matrix, in which *p* is the number of available processors at the time of allocation and *t* is the number of tasks. The position matrix of each particle has the two following properties:

1) All the elements of the matrices have either the value of 0 or 1. In other words, if X_{id} is the position matrix of i-th particles in a d-dimensional space , then: $X_{id}(t,p) \in (0,1)$
2) In each row of these matrices only one element is 1 and others are 0.

In position matrix each row represents a task allocation and each column represents allocated tasks in a processor. Velocity V_{id} of each particle is considered as a $t \times p$ matrix whose elements are in range $[-V\max, V\max]$. Also Pbest and nbest are $t \times p$ matrices and their elements are 0 or 1 as position matrices. p_{id} represents the best position that i-th particle has visited since the first time step and p_{gd} represents the best position that *i*-th particle and its neighbors have visited from the beginning of the algorithm. In this paper we used star neighborhood topology for p_{gd}. In each time step p_{id} and p_{gd} should be updated:

$$V_{id}^{new} = weight \times V_{id}^{old} + C_1 \times rand_1 \times (p_{id} - X_{id}) + C_2 \times rand_2 \times (p_{gd} - X_{id}) \quad (6)$$

$$X_{id}^{new}(t,p) = \begin{cases} 1 & if\ V_{id}^{new}(t,p) = \max\{V_{id}^{new}(t,p)\} \\ 0 & otherwise \end{cases} \quad (7)$$

In (6) $V_{id}^{new}(t,p)$ is the element in *t*-th row and *p*-th column of the *i*-th velocity matrix in the updated time step of the algorithm and $X_{id}^{new}(t,p)$ denotes the element in t-th row and *p*-th column of the *i*-th position matrix in the updated time step. C_1 and C_2 are positive acceleration constants which control the influence of P_{id} and P_{gd} on the search process. Also $rand_1$ and $rand_2$ are random values in range [0, 1] sampled from a uniform distribution. *weight* which is called inertia weight was introduced by Shi and Eberhart [7] as a mechanism to control the exploration and exploitation abilities of the swarm. Usually w starts with large values (e.g. 0.9) which decreases over time to smaller values so that in the last iteration it ends to a small value (e.g. 0.1).
Eq. (7) means that in each row of position matrix value 1 is assigned to the element whose corresponding element in velocity matrix has the max value in its corresponding row. If in a row of velocity matrix there is more than one element with *max* value, then one of these elements is selected randomly and 1 assigned to its corresponding element in the position matrix.

The pseudo code of the proposed DPSO algorithm is stated as follows:

```
Create and initialize a $t \times$p -dimensional swarm with $P$ particles
repeat
for each particle i=1,$\cdots$,P do
  if $f(X_{id}) > f(p_{id})$ then  // f() represent the fitness function of Eq.(1)
      $P_{id} = X_{id}$ ;
  end
  if $f(P_{id}) > f(P_{gd})$ then
      $P_{gd} = P_{id}$ ;
  end
end
  for each particle i=1,$\cdots$,P do
      update the velocity matrix using Eq. (6)
      update the position matrix using Eq. (7)
  end
until stopping condition is true;
```

4. Experimental results

In this section, we will present the experimental results and comparative the computational performance. The platform for conducting the experiments in a PC with Dual Core Processor 4400+2.29 GHz CPU and 1.75GB RAM. All programs are coded in Java programming language in Borland JBuilder 2006.

We give a formal description of our tasks allocation model. We start with a description of a cloud infrastructure. Then, we formalize user tasks and allocation of tasks on the cloud infrastructure. In our model, we represent a cloud as a connected graph of networked computation nodes. We assume that there exists a communication link between each pair of nodes. We also assume that each link has an individual bandwidth and the data transfer on one link does not affect the other links. A node n corresponds to a computing entity like a physical or a virtual machine. An edge e is a communication link between two nodes.

Figure 2 shows an example of a cloud. The cloud is depicted by the directed acyclic graph (DAG). The nodes contain tasks by users submit to be executed on the cloud. The upper part of the node , ec , represent task execution cost. The numbers on the edges represent the communication cost of bandwidth links.

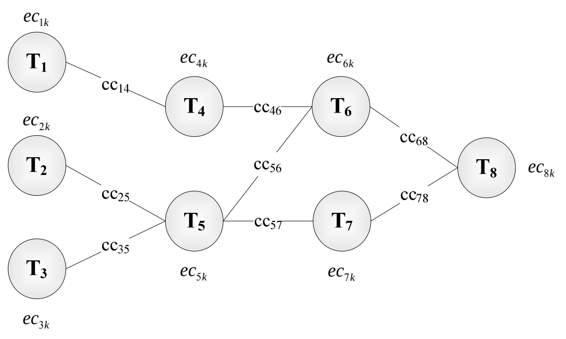

Figure 2 the directed acyclic graph of tasks

To simulate our proposed DPSO algorithm for interconnection tasks graph in figure 2 , we have used the data set of Amazon EC2 Standard Instance are shown in Table 1. The stopping criterion in DPSO is the number of generations such that no improvement is obtained in the value of fitness function (figure 3).The achieve results of eight tasks allocation are shown in Table 2.

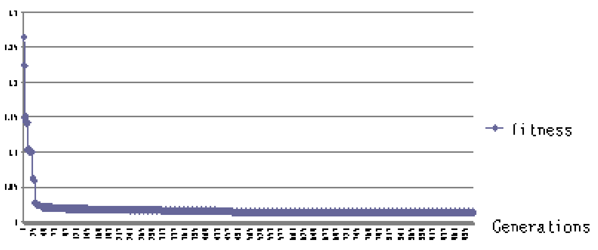

Figure 3 The convergence of DPSO for eight tasks allocation

Table1. Amazon EC2 Standard Instance

Processors		Memory(M)	CPU(GB)	Executed cost(ec)
EC2 Standard Instance	P_1	1.7GB	8.0GB~9.6GB	\$0.12~\$0.14
	P_2	1.7GB	8.0GB~9.6GB	\$0.12~\$0.14
	P_3	15GB	8.0GB~9.6GB	\$0.96~\$1.11
	P_4	7.5GB	5.0GB~6.0GB	\$0.48~\$0.52
	P_5	7.5GB	5.0GB~6.0GB	\$0.48~\$0.52
	P_6	1.7GB	4.0GB~4.8GB	\$0.12~\$0.14
	P_7	1.7GB	4.0GB~4.8GB	\$0.12~\$0.14
	P_8	1.7GB	4.0GB~4.8GB	\$0.12~\$0.14

4.1 Comparative performances

In this section, we present the comparative performances between the proposed DPSO and mathematical programming(Table 3). The parameter values used in both of DPSO and mathematical programming LINGO are optimally tuned by intensive preliminary experiments to let the competing algorithms perform at the best level. To be specific, the parameter setting used by DPSO is (number of particles=15, c1=1,c2=3) and $cc_{ij} = 0.000208$.

Table 3. Comparison of the performance for various tasks allocation

Quantity		Heuristics		Math. programming	
		DPSO		LINGO	
Tasks	Processors	fitness	Time (sec)	Min Cost	Time (sec)
4	4	0.507	0	0.507	0
8	8	1.013	0.469	1.011	1
12	12	2.944	1.234	2.936	16

.

Table 2 The eight tasks allocation solutions through DPSO

Optimal Allocation	Task 1	Task 2	Task 3	Task 4	Task 5	Task 6	Task 7	Task 8
	Processor 8	Processor 7	Processor 7	Processor 1	Processor 2	Processor 2	Processor 7	Processor 6
Cost	Total execution cost and communication cost \$1.011							

5. Conclusions

This paper presented a version of Discrete Particle Swarm Optimization (DPSO) algorithm for tasks allocation. We used the heuristic to minimize the total cost of application tasks excution on Cloud computing environments. The performance of the proposed algorithm was compared with the mathematical programming method through carrying out exhaustive simulation tests and different settings. Experimental results show that the advantage of the DPSO algorithm is its speed of convergence and the ability to obtain faster and feasible allocation. As future work, the authors of the paper plan to carry out extended simulation studies that consider not only CPU time and memory space share but also network bandwidth as resources.

Acknowledgments

This research work was sponsored by the National Science Council, R.O.C., under project number NSC99-2221-E-155-022.

References

[1] W. Chung, R. Chang, "A new mechanism for resource monitoring in Grid computing", *Future Generation Computer Systems* ,Vol. 25. No.1.,2009,pp. 1-7.
[2] M. Armbrust, A. Fox, R. Griffith, A. D. Joseph, R. H. Katz, A. Konwinski, G. Lee, D. A. Patterson, A. Rabkin, I. Stoica, and M. Zaharia. "Above the Clouds: A Berkeley View of Cloud Computing" . *Technical Report UCB/EECS-2009-28*, EECS Department, University of California, Berkeley, Feb 2009.
[3] I. Foster, Y. Zhao,I. Raicu, S. Lu, S." Cloud computing and grid computing 360-degree compared", *Grid Computing Environments Workshop*,2008, pp. 1–10.
[4] Amazon Elastic Compute Cloud, http://aws.amazon.com/ec2
[5] A. Ernst, H. Hiang, M. Krishnamoorthy," Mathematical programming approaches for solving task allocation problems", *Proc. of the 16th National Conf. Of Australian Society of Operations Research*, 2001.
[6] G.H. Chen, J.S. Yur, "A branch-and-bound-with-derestimates algorithm for the task assignment problem with precedence constraint", Proc. of the 10th International Conf. on Distributed Computing Systems, 1990, pp. 494– 501.
[7] Zs. Németh, V. Sunderam, "Characterizing grids: Attributes, definitions, and formalisms", *Journal of Grid Computing,*Vol. 1 . No.1,2003,pp. 9-23.
[8] A. Abraham, H. Liu, M. Zhao," Particle swarm scheduling for work-flow applications in distributed computing environments, in: Metaheuristics for Scheduling: Industrial and Manufacturing Applications," in: Studies in Computational Intelligence, Springer Verlag, Germany, 2008, pp. 327-342.
[9] A. Abraham, R. Buyya, B. Nath, "Nature's heuristics for scheduling jobs on computational Grids", in: Proceedings of the 8th International Conference on Advanced Computing and Communications, Tata McGraw-Hill, India, 2000, pp. 45-52.
[10] Y. Gao, H.Q. Rong, J.Z. Huang," Adaptive Grid job scheduling with genetic algorithms", *Future Generation Computer Systems*,Vol. 21,No. 1,2005, pp. 151-161.
[11] A. Abraham, R. Buyya and B. Nath, "Nature's heuristics for scheduling jobs on computational grids", *Proc. of the 8th IEEE International Conference on Advanced Computing and Communications*, India, 2000,pp.45-52.
[12] H. Liu, A. Abraham," An hybrid fuzzy variable neighborhood particle swarm optimization algorithm for solving quadratic assignment problems", *Journal of Universal Computer Science*, Vol.13, No.7, 2007, pp. 1032-1054.
[13] P.Y.Yin, S.S. Yu,P.P. Wang, and Y.T. Wang," A Hybrid Particle Swarm Optimization Algorithm for Optimal Task Assignment in Distributed System", *Computer Standards & Interfaces*, Vol. 28, 2006, pp. 441-450.
[14] P. Ruth, X. Jiang, D. Xu, and S. Goasguen. "Virtual distributed environments in a shared infrastructure". *Computer*, Vol. 38, No. 5, 2005,pp.63–69.

Yee Ming Chen is a professor in the Department of Industrial Engineering and Management at Yuan Ze University, where he carries out basic and applied research in agent-based computing. His current research interests include soft computing, supply chain management, and system diagnosis/prognosis.

Shin-Ying Tsai was a graduated student in the Department of Industrial Engineering and Management at Yuan Ze University, where she was studying basic and applied research in Cloud computing and heuristic algorithms. She now works in Gold Circuit Electronics as a Design Engineering.