Glowworm Swarm Optimisation Based Task Scheduling for Cloud Computing

Dabiah Ahmed Alboaneen
Department of Computer, Communications and Interactive Systems
Glasgow Caledonian University
Glasgow, UK
dabiah.alboaneen@gcu.ac.uk

Huaglory Tianfield
Department of Computer, Communications and Interactive Systems
Glasgow Caledonian University
Glasgow, UK
h.tianfield@gcu.ac.uk

Yan Zhang
Department of Computer, Communications and Interactive Systems
Glasgow Caledonian University
Glasgow,UK
yan.zhang@gmail.com

A. *ABSTRUCT*

Task scheduling is a non-deterministic polynomial-time hard(NP-hard) optimisation problem, thus applying metaheuris-tics is important. In this paper, we employ glow-worm swarm optimisation (GSO) to solve the task scheduling problem in cloud computing to minimise the total execution cost of tasks while keeping the total completion time within the deadline. Simulation results show that GSO based task scheduling (GSOTS) algorithm outperforms shortest task first (STF), largest task first (LTF) and particle swarm opti-misation (PSO) algorithms in reducing the total completion time and the cost of executing tasks.

B. *CCS Concepts*

•Computer systems organization → Cloud comput- ing;

C. *KEYWORDS*

Cloud computing; resource management; glowworm swarm optimisation (GSO); task scheduling

I. INTRODUCTION

1. INTRODUCTION Task scheduling is crucial in cloud computing, which is a process that maps users' tasks to suitable resources in the form of virtual machines (VMs) to execute.There are two types of task scheduling in cloud computing, i.e., (i) Static task scheduling where all tasks arrive at the same time and they are known a priori to scheduling. In this case, tasks are assigned to VMs in a static way. (ii) Dynamic task scheduling where all the tasks are scheduled instantly once they arrive. When users' tasks need to be scheduled, users usually sign a service level agreement (SLA) with cloud providers. In this SLA, the quality of service (QoS) requirements of the task scheduling should be clearly defined such as, the dead- line of each task, task scheduling budget and service secu-rity. Each cloud user has to decide which and how many VMs need to be provisioned before actually requesting and paying for them. Moreover, as cloud resources are pay-per-use, the cost of task execution has to be taken into account. So, task scheduling directly affects the performance of cloud computing.Task scheduling in cloud computing can be modelled as a bin-packing problem and is a non-deterministic polynomial-time hard (NP-hard) problem [1]. This problem becomes more challenging with the increase of the complexity of cloud computing. Generally, it is difficult to develop algorithms for producing optimal solutions within a short time. Recently,using metaheuristic algorithms such as genetic algorithm (GA) and particle swarm optimisation (PSO) to deal with cloud resource scheduling has received increasing attention due to the ability of the algorithms to provide near-optimalsolutions within a reasonable time [2].To address the aforementioned problem, this paper pro-poses a solution by employing glowworm swarm optimisa-tion (GSO) for improving the execution cost of scheduled independent tasks. The execution cost of a task considers completion time and the price of using the VM. Completion time includes waiting time and execution time of the task and it should be within the user defined deadline to meet the user requirements. To evaluate the performance of the proposed GSO based task scheduling (GSOTS) algorithm, we undertake a comparative simulation study of the proposed algorithm with shortest task first (STF), largest task first(LTF) and PSO algorithms.The

ICC '17, March 22 2017, Cambridge, United Kingdom c 2017 ACM. ISBN 978-1-4503-4774-7/17/03. . . $15.00DOI : http : //dx.doi.org/10.1145/3018896.3036395

VM type	MIPS	Pe	Capacity	price($/Hours$)
Type 1(sml)	500	4	2000	0.34
Type 2(mid)	1000	7	70000	0.5
Type 3(lrg)	1500	20	30000	0.6

TABLE I
AMAZON EC2 INSTANCE TYPES AND PRICES

remainder of this paper is arranged as follows. Sec-tion 2 gives the task scheduling problem formulation. The GSOTS algorithm is put forward in Section 4. Simulations setup and the results are presented in Section 5. Section 6 discusses the work related to the scheduling algorithms in a cloud environment Finally, Section 7 draws the conclusion and points out the future work.

II. 2. PROBLEM FORMULATION

The general system is depicted in Figure 1. Assume that a cloud platform is supported by physical machines (PMs) PM1, .., PMP , each of which hosts a set of VMs via the corresponding virtual machine monitor (VMM) and each PM runs resource manager (RM) which is responsible for monitoring the resource utilisation of PMs and collecting run-time statistics of each PM, including PM availability, resource utilisation and VM status. There are M independent tasks need to be scheduled. Tasks are submitted to the computing system (1). The task scheduler collects the feedback information from the RMs to maintain an overall view of the system resource utilisation as marked by information (2). Based on the information of user requests received from the service provider and the feedback received from RMs, the task scheduler initiates the task scheduling algorithm to schedule tasks on VMs (3) and VM scheduler aims to allocate VMs on PMs as marked by information (4). In our task scheduling algorithm, the task scheduler first calculates the completion time to execute the task which is based on the execution time and waiting time of task. The execution time is calculated based on the number of instructions in each task which is received from the user side and the VM capacity which is received from the RM side. If the completion time of executing task is within the user defined deadline, then the task can be executed on the VM. After that, the task scheduler calculates the execution cost of task in each available VM. To minimise the execution cost of the task, task scheduler decides which VM is suitable to be provisioned to each task to meet the requirements of each task. Finally, the task scheduler will return the result of execution to user when all tasks are done.

A. Assumption

(i) Tasks are independent of each other.

(ii) Each task is allowed to be processed on any given available VM that meets the requirement.

(iii) The execution time of each task is VM-dependent.

(iv) Each task must be completed without interruption once started (non-preemptable).

(v) Each VM can be provisioned to more than one task.

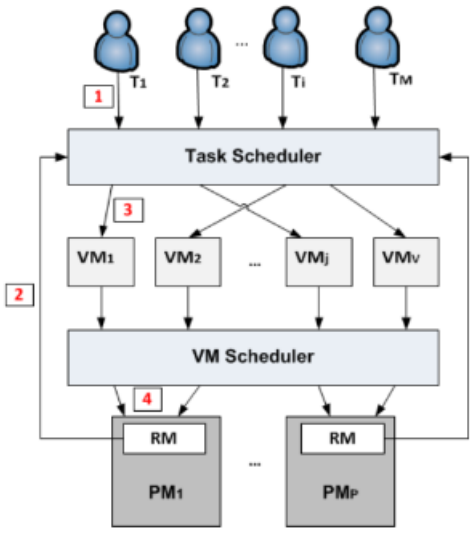


Fig. 1. Figure 1. Task and VM scheduling in cloud computing

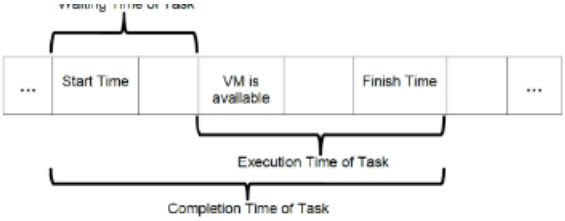


Fig. 2. Caption

B. Inputs:

• Set of tasks is defined as T = T1, ..., Ti, ..., TM, where i [1, M] and M is the total number of tasks. Each task i is described as Ti(insti, deadlinei, arvi), where insti, deadlinei and arvi represent the task size that is measured by million of instructions (MI), task deadline and task arrival time, respectively. • Set of VMs is defined as V M = V M1, ..., V Mj , ..., V MV , where j [1, V] and V is the total number of VMs. Each V Mj is described as V Mj (Cj , P ricej), where Cj is the VM capacity which is the processing speed and expressed in terms of million instructions per second (MIPS). P ricej is the amount spent for using a VM in an hour. We consider three types of VMs offered by the cloud provider to the user as vmsml, vmmed, vmlrg, and each VM type has different

C. Output:Selected VM for each task.

D. Objective: The aim of the GSOTS algorithm is to schedule tasks to appropriate VMs in order to minimise the execution cost (ECij) as below.

$$minf = \sum_{i=1}^{M} EC_ij \quad (1)$$

E. Execution Cost:

The execution cost (ECij) of task i is defined as multiplication of the price of V Mj and the completion time of task i, that is,

$$EC_ij = Price * CT_ij \quad (2)$$

Be sure that the symbols in your equation have been defined before or immediately following the equation. Use "(**??**)", not "Eq. (**??**)" or "equation (**??**)", except at the beginning of a sentence: "Equation (**??**) is . . ."

F. Basic Algorithm

GSO algorithm is a swarm intelligence algorithm developed by Krishnan and Ghose. GSO is a population-based algorithm. As control is not centralised at a single point, it is more scalable [3]. In our earlier work [4], GSO algorithm has been applied for dynamic VMs allocation in cloud computing.

GSO is based on the behaviour of glowworms. A glowworm that produces more light (high luciferin) means that it is closer to an actual position and has a high objective function value. Each glowworm selects a neighbour that has a higher luciferin value than its own, according to a probabilistic estimation, and moves towards it. These movements are based only on local information. Therefore, it enables the glowworms to divide into subgroups leading to the detection of multiple optima of the given objective function.

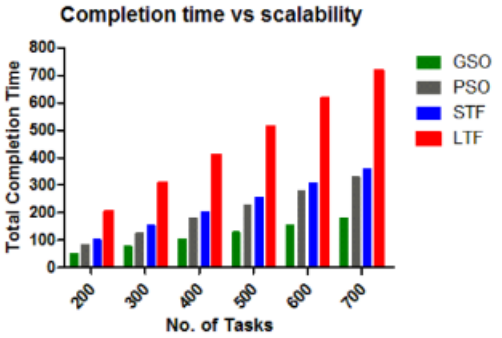
GSO algorithm starts by positioning glowworms randomly in the workspace and all the glowworms contain an equal quantity of luciferin. A GSO algorithm comprises four phases, i.e., initialisation, luciferin updating, moving and local radial range updating. The GSO algorithm is shown in Algorithm

G. *The initial population consists of number of glowworms, which are considered as candidate solutions to the problem. The initial population is generated randomly. Each V Mj has a location xj (t), that is defined by the completion time that VM takes it to execute the task, a luciferin value 'j (t), a local radial range j d (t), max sensor range s, the size of moving step s, the number of desired neighbours nt, the luciferin decay coefficient p (0 ¡ p ¡ 1), the luciferin enhancement coefficient and the change rate of the neighbourhood range*

Algorithm 2: GSOTS: GSO Based Task Scheduling Input: vmList, taskList Output: selectedV m 1 Initialise parameters , p, s, nt, j d

2 t = 1 3 for i taskList do 4 taski → vmj randomly 5 for j vmList do 6 'j (t) = P ricej CTij 7 for i taskList do 8 while termination condition not met do 9 for j vmList do 10 'j (t) = updateLuc(j) by Eq. (6) 11 Nj (t) = getV mNeighbours(j, nt) by Eq. (7) 12 for n Nj do 13 pjn(t) = calcP rob() by Eq. (8) 14 Select the highest pjn(t) 15 selectedV m ← highest pjn(t) 16 Update j d by Eq. (10)

17 t = t + 1 18 if selectedV m 6= NULL then 19 return selectedV



(a) Total completion time in the static workload

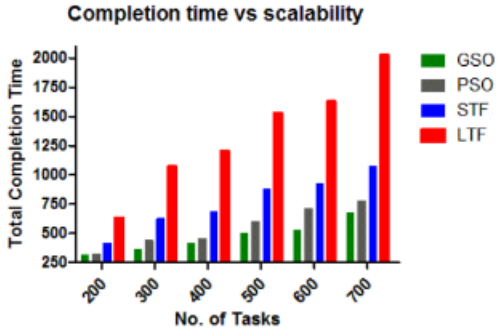


Fig. 3. Caption

parameter	p	b	r	n_t	s	l_o
value	0.4	o.6	0.8	0.9	0.10	

TABLE II
CAPTION

H. 5. SIMULATION

REFERENCES

I. Resource Utilisation

The load of all VMs can be calculated by using the information that is received from the data centre. Thus, stan- dard deviation (SD) is calculated to measure the deviations of tasks' load on VMs. SD of load can be defined as below.

$$X = \frac{\sum_{j=0}^{v} CT_ij - x}{\sum_V} \quad (3)$$

J. Conclusion

We have applied GSO algorithm to solve the task scheduling problem in cloud computing. The proposed algorithm aims to minimise the execution cost of task using Amazon EC2 pricing model under deadline constraint. The simulation results have shown that GSOTS outperforms PSO, STF and LTF algorithms in terms of minimising the total completion time when scheduling different number of tasks and VMs in cloud. In addition, GSOTS can utilise the resources more efficiently than other algorithms. Regarding the cost, GSOTS is able to reduce the execution cost more than PSO algorithm. Extending the GSOTS algorithm to consider dynamic task scheduling is one of our future work..

REFERENCES

[1] [1] T. A. Genez, L. F. Bittencourt, and E. R. Madeira. Workflow scheduling for saas/paas cloud providers considering two sla levels. In IEEE Network Operations and Management Symp.,(Maui,16-20 Apr.), pages 906-912. IEEE, 2012.

[2] [2] B. Jennings and R. Stadler. Resource management in clouds: Survey and research challenges. Journal of Network and Sys. Management, 23(3):567-619, 2015..

[3] K. Krishnanand and D. Ghose. Glowworm swarm based optimization algorithm for multimodal functions with collective robotics applications. Multiagent and Grid Sys., 2(3):209-222, 2006..

[4] [4] D. Alboaneen, H. Tianfield, and Y. Zhang. Glowworm swarm optimisation algorithm for virtual machine placement in cloud computing. In Int. IEEE Conf. on Ubiquitous Intelligence Comp., Advanced and Trusted Comp., Scalable Comp. and Communications, Cloud and Big Data Comp., Internet of People, and Smart World Congress,(Toulouse, 18-21 July), pages 808–814. IEEE, 2016..

[5] R. N. Calheiros, R. Ranjan, A. Beloglazov, C. A. De Rose, and R. Buyya. Cloudsim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms.

[6] D. G. Feitelson, D. Tsafrir, and D. Krakov. Experience with using the parallel workloads archive. Journal of Parallel and Dist. Comp., 74(10):2967-2982, 2014.