An improved Adaptive workflow scheduling Algorithm in cloud Environments

Yinjuan Zhang
College of Information Engineering
Yangzhou University
Yangzhou 225009, China
E-mail: 736424972@qq.com

Yun Li *
College of Information Engineering
Yangzhou University
Yangzhou 225009, China
E-mail: liyun@yzu.edu.cn

***Abstract*—Cloud environments consist of a collection of tasks and heterogeneous resources, the problem of mapping tasks to resources is a major issue. In this paper, we proposed an Improved Adaptive heuristic algorithm (IAHA). At first, the IAHA algorithm makes tasks prioritization in complex graph considering their impact on each other, based on graph topology. Through this technique, completion time of application can be efficiently reduced. Then, it is based on adaptive crossover rate and mutation rate to cross and mutate to control and lead the algorithm to optimized solution. The experimental results show that the proposed method solution can obtain the response quickly moreover optimize makespan, load balancing on resources and failure rate of tasks. At the same time, the proposed algorithm presents the better results contrast with traditional genetic algorithm (GA) and HSGA.**

Keywords—Cloud environments; Adaptive genetic algorithm; Task scheduling

I. INTRODUCTION

Cloud computing is the further development of parallel algorithm, distributed computing and grid computing. It includes a collection of heterogeneous computing nodes, virtualized computers, and software services that are dynamically provisioned. These software services compete end-user's application by their availability, performance, capability, and Quality of Service (QoS) requirements. The cloud computing provisions service in main three levels: Software as a Service (SaaS), Platform as a Service (PaaS), and Infrastructure as a Service (IaaS) [1].In IaaS level, it takes Virtual Machines (VMs) as scheduling units for mapping physical heterogeneous resources to tasks. Each VM is an abstract unit which has computing and storage capacities in cloud. Because of heterogeneous and dynamic properties of resources, in addition to many numbers of entry tasks with different characteristics, this issue of task scheduling is known to be a NP-Complete problem. The software running in cloud environments is called an application. Each application is a workflow that consists of a set of tasks, which are connected to each other. These tasks would be scheduled to run over different processors in cloud environments.

Many methods are proposed for this problem. Some methods have already been used in the scheduling of workflows [2, 3]. But they don't consider heterogeneous components characteristics and are not suitable for cloud computing which is heterogeneous distribute computing system. Some proposed algorithms take heterogeneous components characteristics into account [4], but they use simple-structure directed acyclic graph (DAG). The relationships between tasks of workflow in cloud environments are complex, so after workflow converts to DAG graph, the stratification is not obvious. For this reason, the complex-structure DAG graph is more suitable for cloud environment. HSGA algorithm [5] takes heterogeneous of resources in cloud environment into consideration, moreover it uses complex-structure graph to describe workflow, but crossover rate and mutation rate in HSGA are fixed, which is easy to cause premature phenomenon.

This paper mainly targets complex computation-intensive tasks, we improve HSGA algorithm to propose IAHA algorithm. The IAHA algorithm tries to decrease the number of algorithm operation iteration with tasks preprocessing to obtain an optimized initial population, considering the load balance on resources, then execute crossover and mutation operation according to the adaptive crossover rate and mutation rate to control premature phenomenon, thus lead algorithm to optimal solution.

II. RELATED WORK

In this section, we mention simple-structure and complex-structure of DAG graph. In this meantime, we introduce HSGA algorithm.

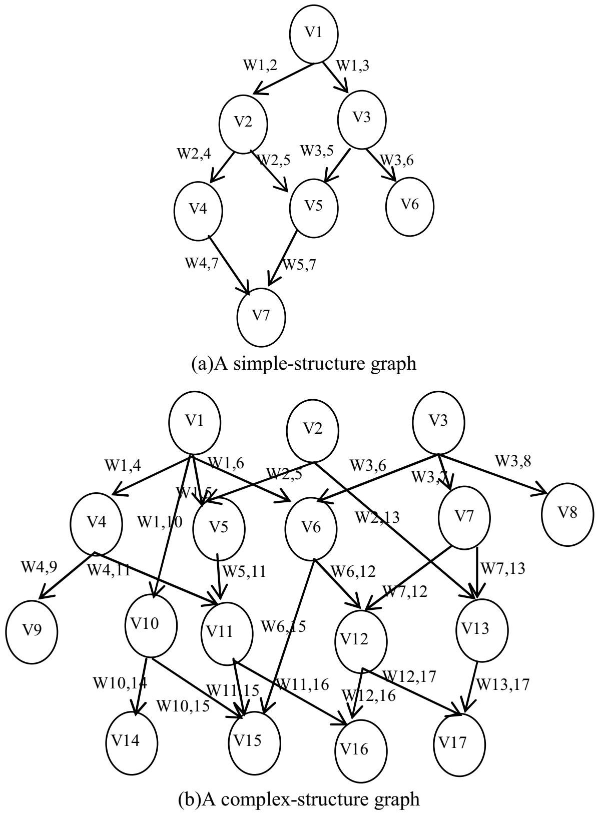
A. Simple structure and complex structure of DAG graph

Each application in cloud environments is a workflow that contains a set of tasks, which tasks are connect to each other by precedence constraints. Each task will be executed and give an output data. This data will be then sent to the next task as defined by the structure of workflow. Most workflow can be represented by a DAG graph whose nodes represent tasks and whose edges represent precedence constraints. According to

978-1-4673-8537-4/15 $31.00 © 2015 IEEE
DOI 10.1109/CBD.2015.27

Authorized licensed use limited to: SOUTH CHINA UNIVERSITY OF TECHNOLOGY. Downloaded on May 09,2020 at 02:41:38 UTC from IEEE Xplore. Restrictions apply.

different workflow projects, the workflow application structures can be classified as either simple structure like Fig. 1(a) or complex structure as Fig. 1(b). In the simple-structure graph, nodes are related together considering certain level, but the complex-structure graph like [6] has more relation among nodes where the level of some nodes is not certain.



(a)A simple-structure graph

(b)A complex-structure graph

Fig. 1. The simple and complex graphs in workflow

B. HSGA algorithm

1) Encoding

In GA algorithm, every solution is encoded as a chromosome. There's a lot of ways to encode chromosome, how to encode that depends on the problem we will to resolve. Fig. 2 depicts a chromosome in HSGA algorithm.

Here, first, the tasks of graph are ordered on priority according to section 2), then map tasks to processors.

2) Task prioritization

Tasks on complex application graph can't be partitioned into level easily, and each task's length and successors are different with others, so the task selection based on graph topology is an important problem. Each task's completion time influences the completion time of application, so executing order of tasks, in other words, an ordering method of tasks has important influence on total completion time. Thus HSGA algorithm computes the priority of tasks based on task influence, according to (1):

V1	V3	V4	V2	V5	• • • •	V9
P2	P9	P3	P1	P8	• •	P25

Fig. 2. A simple encoding of a schedule as a chromosome

$$T_{\text{priority}}(v_i) = w(v_i) + \sum_{d(v_j)=\alpha}^{\beta} (d(v_j) \times w(e_{i,j})), \quad v_j \in T_{pred}(v_i) \tag{1}$$

Where task v_j is one of predecessors of task v_i, $d(v_j)$ is the depth of task v_j in critical path. The critical path for each task is the longest path from it to an exit task. $T_{pred}(v_i)$ is a set of predecessors of task v_i, β is the most depth of successors with the longest sequence and α is computed as:

$$\alpha = \beta - \lfloor \beta/2 \rfloor \tag{2}$$

3) Initial population

HSGA algorithm sorts tasks and puts that in all chromosomes' first line. Then each task is mapped to a selected resource from a virtual list of available resources, according to performance (computational speed, load, etc.) of virtual machines.

4) Evaluation of fitness function

The fitness value of each solution is computed through:

$$\text{Fitness} = \sum_{i=1}^{m} (p_i^t \times p_i^f) \tag{3}$$

Where p_i^t is the time when resource p_i completes all its assigned tasks of workflow in current scheduling, p_i^f is probability of task failure per unit time on resource p_i as failure rate of resource. The chromosome with minimum fitness value is considered as the best solution among the others.

Target is : minimizing(Fitness) (4)

5) Selection, crossover, mutation operation

Some of the best of chromosomes will be selected by elitism method for next iteration. And here a random gene selection crossover is used.

A chromosome and one of its genes will be chosen randomly and a resource will be selected randomly from virtual list of resources in mutation operation. The selected resource will be replaced with selected gene if its failure rate is better than the last candidate resource as [7] and this candidate resource is not the resource with most workload in current schedule. The failure factor of resource can be computed as:

$$p_i^{fr} = p_i^f / p_i^{MIPS} \tag{5}$$

Authorized licensed use limited to: SOUTH CHINA UNIVERSITY OF TECHNOLOGY. Downloaded on May 09,2020 at 02:41:38 UTC from IEEE Xplore. Restrictions apply.

Where p_i^{MIPS} is the computing power of resources as number of machine instructions per second.

III. THE IMPROVED ALGORITHM

This paper mainly focuses on task prioritization, crossover and mutation operation of HSGA algorithm to execute corresponding improvements.

A. Task prioritization

Assume sorting tasks according to priority value of HSGA algorithm directly, it will have the following problems: if parent nodes of the child node is excessive or the edge of these parent nodes have a large value, that will result in a priority value of child node greater than parent nodes, child node perform prior to its parent nodes, thus is not in conformity with the topology sequence of DAG graph.

Based on the above problems, this paper adopts the following sort method to calculate tasks' weight, as shown in (6):

$$\mathrm{T}_{\text{priority}}(v_i) = w(v_i) + \sum_{d(v_j)=\alpha}^{\beta} (d(v_j) \times w(e_{i,j})), \quad v_j \in \mathrm{T}_{\text{succ}}(v_i) \tag{6}$$

Where $\mathrm{T}_{\text{succ}}(v_i)$ is a set of successors of task v_i, representation of α is same as (2). Here, the tasks are ordered by weight in descending then depth in descending. In this method, it will ensure the reasonableness of task execution, but also optimize the total completion time.

B. Crossover and mutation operation

These operations are quite important in genetic algorithm, because these implement the principles of evolution. In this paper, adaptive crossover and mutation rate are adopted to execute crossover and mutation operation. Crossover and mutation rate of HSGA algorithm are determined in advance and remain unchanged in the evolutionary process, which will likely lead to premature, thereby reducing the search efficiency. Literature [8] proposed a new evaluation index to evaluate the premature length of population:

Assume that t_{th} generation of populations consists of M individuals, the average fitness value is $\overline{\mathrm{F}_{\mathrm{t}}}$, the best individual fitness value is $\mathrm{F}_{\mathrm{t}\max}$, $\overline{\mathrm{F}_{\mathrm{t}\max}}$ represent average fitness of which individual fitness value is greater than $\overline{\mathrm{F}_{\mathrm{t}}}$, then:

$$\Delta = \mathrm{F}_{\mathrm{t}\max} - \overline{\mathrm{F}_{\mathrm{t}\max}} \tag{7}$$

Crossover rate is computed as:

$$\mathrm{P}_c = 1/\left(1+\exp(-k_1\Delta)\right) \tag{8}$$

Mutation rate is computed as:

$$\mathrm{P}_{\mathrm{m}} = 1 - 1/\left(1+\exp(-k_2\Delta)\right) \tag{9}$$

Where $\mathrm{k}_1, k_2 > 0$, because Δ is always greater than or equal to 0, so the range of P_{c} and P_{m} is [0.5,1]. Thus, in the evolutionary process, when individual of population tends to be discrete, that Δ becomes larger, P_{c} increase, and P_{m} decrease, the capabilities of finding best individual will strengthen. When individual of population tends to converge, then Δ becomes smaller, P_{c} decrease, P_{m} increase, the ability of producing new individuals will strengthen.

One point crossover and two point crossover are used in this paper, shown in fig. 3 and fig.4:

Before Crossover:

V1	V3	V4	V2	V5	V7	V8	V6
P1	P3	P5	P7	P9	P11	P15	P20

V1	V3	V4	V2	V5	V7	V8	V6
P2	P4	P12	P8	P13	P6	P10	P14

After Crossover:

V1	V3	V4	V2	V5	V7	V8	V6
P1	P3	P5	P7	P9	P6	P10	P14

V1	V3	V4	V2	V5	V7	V8	V6
P2	P4	P12	P8	P13	P11	P15	P20

Fig. 3. One point crossover

Before Crossover:

V1	V3	V4	V2	V5	V7	V8	V6
P1	P3	P5	P7	P9	P11	P15	P20

V1	V3	V4	V2	V5	V7	V8	V6
P2	P4	P12	P8	P13	P6	P10	P14

After Crossover:

V1	V3	V4	V2	V5	V7	V8	V6
P1	P3	P5	P8	P13	P6	P15	P20

V1	V3	V4	V2	V5	V7	V8	V6
P2	P4	P12	P7	P9	P11	P10	P14

Fig. 4. Two point crossover

C. Pseudo code

Pseudo code of IAHA algorithm

```
Input: Set of available resources and unmapped tasks
Output: An optimized generated schedule
1.  Compute depth of each task in DAG graph
2.  Compute the priority of each task according to formula(6), and sort tasks
    by rule
3.  Update resource properties
4.  Make a list of available resources
5.  For each chromosome do
6.    Find the best fit resources for each task based on execution time
```

Authorized licensed use limited to: SOUTH CHINA UNIVERSITY OF TECHNOLOGY. Downloaded on May 09,2020 at 02:41:38 UTC from IEEE Xplore. Restrictions apply.

```
7.      Jump to next position in resource list
8.      If counter pointer point to the last resource then
9.          Update resource properties
10.         Jump to the first position in resource list
11.     End If
12. End for
13. Select 20 chromosomes to compose a population, and initialize
    crossover and mutation rate
14. While satisfy stop condition
15.     Compute fitness value of every chromosome, adaptively adjust
    crossover rate and
            mutation rate according to formula (7),(8),(9)
16.     Elitism method is used in selection operation
17.     Crossover and mutate to get a new generation of population
18.     If Optimal chromosome in 10 generations is not changed
19.         Double population size
20. End while
21. Save the best solution
22. Dispatch all mapped tasks on candidate resources due to obtain the best
    solution
```

IV. Simulation and Result Analysis

To verify correctness and rationality of the algorithm results, this section presents the comparative evaluation IAHA with traditional GA and HSGA algorithm, and evaluates three subjects: makespan of application, failure rate and load balance of resource. In this paper, we use experimental simulation platform named CloudSim, the simulation parameters are listed in Table 1.

TABLE I. Simulation Parameters

Parameter	Value
Number of tasks in application	30-100
The number of resources	30
Task length	1200000-7200000(MIPS)
Failure rates of tasks	0.0001-0.001
Resource Speed	500-1000（MIPS）
Bandwidth between resources	10-100（mbps）

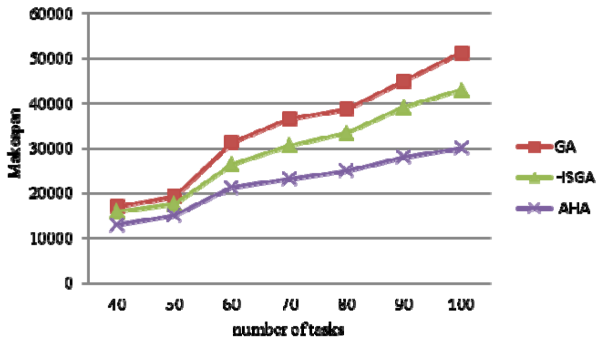


Fig. 5. Makespan for application with increasing number of tasks

In fig.5, other algorithms are compared with improved one. The results show, the completion time of application (makespan) for IAHA is less than GA and HSGA. Also load balance of resources is demonstrated in fig.6. With the growth of the number of tasks, load balance of resources in IAHA algorithm is significantly reduced compared with the other two algorithms. Fig.7 shows that failure frequency of IAHA algorithm and HSGA algorithm is fairly, but compared with GA algorithm is still a relatively large improvement.

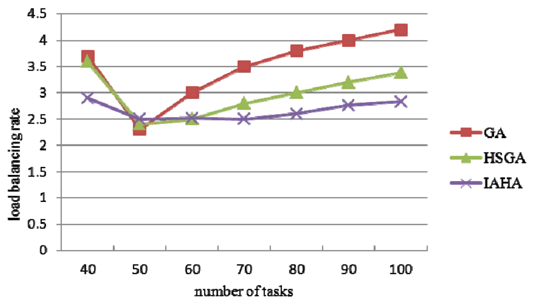


Fig. 6. The load balancing with increasing number of tasks

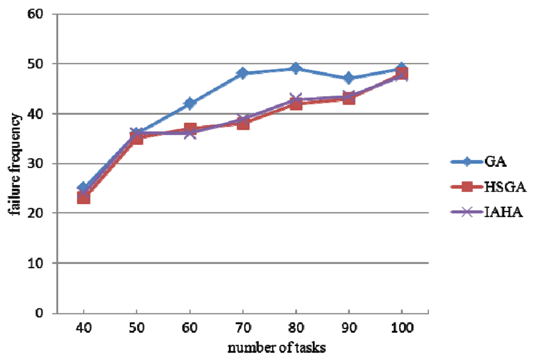


Fig. 7. The failure frequency of algorithms in different number of tasks

V. Conclusions and Future Work

An improved adaptive heuristic algorithm (IAHA) is proposed in this paper. This algorithm focus to process tasks with complex computation which are running in cloud environment, using the algorithm to map resources to tasks reasonably, thereby give a quickly and effectively response to tasks which are users submitted and handled .

In this paper, we use adaptive crossover and mutation rate, but its adaptive extend is not prominent. For example, now we only set values of k_1, k_2 based on experience. In future work, we want to set values of k_1 and k_2 by characteristics of population. In the meantime, the algorithm proposed in this paper is mainly for complex compute-intensive applications, we hope to find a reasonable method to map tasks to resources in case of communication-intensive applications or compute-intensive and communication-intensive mixing in the future.

Acknowledgment

Authorized licensed use limited to: SOUTH CHINA UNIVERSITY OF TECHNOLOGY. Downloaded on May 09,2020 at 02:41:38 UTC from IEEE Xplore. Restrictions apply.

This research was supported in part by the Chinese National Natural Science Foundation under grant No. 61402396, Natural Science Foundation of Jiangsu Higher Education Institutions under contract 13KJB520027, 13KJB510039, The high-level talent project of "Six talent peaks" of Jiangsu Province under contract 2012-WLW-024, Joint innovation fund project of industry, education and research of Jiangsu Province (prospective joint research) under contract BY2013063-10, the talent project of “Green yangzhou and golden phoenix” under contract 2013-50 and graduate student innovative research projects of Yangzhou University under contract CXLX_1415.

References

[1] Ghorbannia Delavar,A.,Aryan,Y.:A Synthetic heuristic algorithm for independent task scheduling in cloud systems.Int.J.Comput.Sci.Issues 2011,8(6),289-295

[2] Yoo,M. :Real-time task scheduling by multi objective genetic algorithm.J.Syst.Softw.2009,Vol 82,619-628

[3] Omara,F.A.,Arafa,M.M.:Genetic algorithms for task scheduling problem.J.Parallel Distrib.Comput.2010,vol.70,13-22

[4] Li Jinmei, Sun Dongwei, Wu Yanxia, Global comparatively optimum static task scheduling algorithm (in Chinese), Application Research of Computes: 2014,Vol.31 No.4, 1027-1030

[5] Ghorbannia Delavar, A., Aryan, Y.: HSGA: a hybrid heuristic algorithm for workflow scheduling in cloud systems. Cluster Comput, 2014, Vol.17:129-137

[6] Ghorbannia Delavar, A., Aghazarian, V., Litkouhi, S., Khajeh Naeini, M.: A scheduling algorithm for increasing the quality of the distributed systems by using genetic algorithm. Int. J. Inf.Educ. Technol.2011, 1(1), 58–62

[7] Wang, X., Yeo, C.S., Buyya, R., Su, J.: Optimizing the makespan and reliability for workflow applications with reputation and a look-ahead genetic algorithm. Future Gener. Comput. Syst.2011, Vol. 27,1124–1134

[8] Li Xin, The Improvement of the Adaptive Genetic Algorithm and its Application (in Chinese),NanJing University of Information Science&Technology, May 2008,32-34

Authorized licensed use limited to: SOUTH CHINA UNIVERSITY OF TECHNOLOGY. Downloaded on May 09,2020 at 02:41:38 UTC from IEEE Xplore. Restrictions apply.