Independent Tasks Scheduling in Cloud Computing via Improved Estimation of Distribution Algorithm

Haisheng Sun [1, a)], Rui Xu [2, b)] and Huaping Chen [1, 3, c)]

[1]*School of Computer Science and Technology, University of Science and Technology of China, Hefei, 230026, China*
[2]*School of Business, Hohai University, Nanjing, 211100, China*
[3]*School of Management, University of Science and Technology of China, Hefei, 230026, China*

[a)]Corresponding author: haisheng@mail.ustc.edu.cn
[b)]rxu@hhu.edu.cn
[c)]hpchen@ustc.edu.cn

Abstract. To minimize makespan for scheduling independent tasks in cloud computing, an improved estimation of distribution algorithm (IEDA) is proposed to tackle the investigated problem in this paper. Considering that the problem is concerned with multi-dimensional discrete problems, an improved population-based incremental learning (PBIL) algorithm is applied, which the parameter for each component is independent with other components in PBIL. In order to improve the performance of PBIL, on the one hand, the integer encoding scheme is used and the method of probability calculation of PBIL is improved by using the task average processing time; on the other hand, an effective adaptive learning rate function that related to the number of iterations is constructed to trade off the exploration and exploitation of IEDA. In addition, both enhanced Max-Min and Min-Min algorithms are properly introduced to form two initial individuals. In the proposed IEDA, an improved genetic algorithm (IGA) is applied to generate partial initial population by evolving two initial individuals and the rest of initial individuals are generated at random. Finally, the sampling process is divided into two parts including sampling by probabilistic model and IGA respectively. The experiment results show that the proposed IEDA not only gets better solution, but also has faster convergence speed.

Key words: Independent Tasks; Scheduling; Improved Estimation of Distribution Algorithm; Genetic Algorithm; Cloud Computing.

INTRODUCTION

Cloud Computing, as an emerging and rapidly evolving IT, has become affected the development of industry and academia more and more widely. With a new dimension of computing resources in the form of Infrastructure as a Service (IaaS), Platform as a Service (PaaS), and Software as a Service (SaaS), cloud computing allows users to purchase virtual resources in the pay-per-use model [1]. In order to balance the interests of users and suppliers, finding an "optimal" scheduling solution is the most important issue in cloud computing resource scheduling. This article focuses on cloud computing resource scheduling on SaaS layer.

Different users will have various Quality of Service (QoS) requirements such as the makespan, user costs and application performance. The objective of this article is to minimize makespan for scheduling independent and indivisible tasks. The literature [2] has proved that such discrete resource scheduling problem is an NP-hard problem, so using evolutionary computation approaches to tackle cloud computing resource scheduling problem has drawn the attention of researchers. Some related research works are summarized as follows: In literature [3], to solve the same problem studied in this paper, the Max-Min and Min-Min methods are merged in standard genetic algorithm (GA) ,but the comparative experimental part is not sufficient. To improve Max-Min, [4] proposed an enhanced Max-Min (EMM) algorithm that select task with average or nearest greater than average execution time firstly, afterwards, a GA based scheduling scheme in which initial population is generated by EMM is applied in literature [5].

Due to that there is no state-of-the-art algorithm and there are deficiencies in the existing works, the estimation of distribution algorithm (EDA) [6] is selected to try tackling the investigated problem. Inspired by [7] which use PBIL to schedule independent and divisible tasks, an IEDA based on PBIL and GA is proposed in this paper by introducing some improvements such as encoding scheme, probabilistic model, adaptive learning rate, initialling population and sampling strategy.

The remainder of this paper is organized as follows: Section 2 describes the problem in detail. Section 3 presents the design of IEDA. The comparative experiment is provided in Section4 and section 5 concludes this paper.

PROBLEM FORMULATION

The problem studied in this paper is to minimize makespan for scheduling independent and indivisible cloudlets on SaaS layer. Our purpose is to find an "optimal" mapping relationship between tasks and resources so that the objective function is optimized. This article considers virtual machines as resources and cloudlets as tasks, respectively. In this paper, the following assumptions are taken into account in the problem investigated.

(1) This paper assumes that all n cloudlets waiting to be processed are independent and cannot be subdivided. At the start of scheduling cloudlets, the number of available VMs in a resource pool is a fixed number m, and all VMs have the same preparation time equal to zero.

(2) The processing speed of each VM Pj (j=1, 2,…, m) and the size of every cloudlets Si (i=1,2,…,n) are foregone. In this paper, we use the million instructions per second (MIPS) to measure the VM processing capacity, and the instruction count (IC) is used to indicate the length of the cloudlet.

(3) The policy that determines the share of processing power among cloudlets in every VM which has single-core is space-shared.

(4) The objective in this paper is to minimize makespan. A feasible solution of the investigated problem is decision vector X=(x1, x2,…,xn), here, xi indicates the assigned resource index of cloudlet i.

Through the above description, the mathematical formulation of the problem in this paper is described as follows:

$$minsize \quad makespan \tag{1}$$

Subject to:

$$x_i \in \{1,2,...,m\}, i = 1,2,...,n \tag{2}$$

$$APT_i = {S_i}/{P_k}, k = x_i \tag{3}$$

$$MP_j = \sum_{z=1}^{w} PT_z, j = 1,2,...,m \tag{4}$$

$$makespan = \max\{MT_j\}, j = 1,2,...,m \tag{5}$$

Constraint (1) presents that the objective is to minimize makespan. Constraint (2) indicates that each cloudlet i has m resource allocation schemes. Constraint (3) ensures that the actual processing time of each cloudlet i APTi depends on the resources arranged. Constraint (4) introduces that the completion time for each VM j MPj is equal to the completion time of the last cloudlet in the VM processing queue, where w represents the total number of cloudlets assigned to VM j, and constraint (5) gives the total completion (excitation) time.

THE DESIGN OF IEDA

In this section, we first introduce the basic framework of PBIL, followed by representing the improvements of PBIL in the proposed IEDA.

Basic Framework of PBIL

The PBIL proposed for variable irrelevant problems is the earliest EDA. The original PBIL is to solve the binary coding optimization problem. The probability model of solution space is represented by a probability vector which can be expressed as $P(X) = (p(x_1), p(x_2), ..., p(x_n))$ .The basic framework and evolutionary progress of PBIL is as follow:

Step1: Generate N individuals as the initial population at random;

Step2: Calculate the fitness of all individuals, and select the best M (M<N) individuals as dominant population;

Step3: Use these M dominant individuals update the probability model P(X) .The Heb rule [8] in machine learning is applied as a criterion to update the probability vector. Here, P1 (X) denotes the probability model of the lth generation, X_l^k represents the kth dominant individual selected by the lth generation, and α denotes learning rate

$$P_{l+1}(X) = (1-\alpha)P_l(X) + \alpha \frac{1}{N} \sum_{k=1}^{N} X_l^k \tag{6}$$

Step4: Sample N new individuals according to the updated probability model P(X)

Step5: If the termination condition is met, return the optimal solution, otherwise, go back to Step2 and continue iterative optimization.

The Improvements of PBIL Algorithm

In this subsection, the improvements of PBIL will be elaborated, mainly including the following aspects: encoding scheme, learning rate, probabilistic model, initialling population and sampling strategy.

Aspects of Improvement

Firstly, the traditional PBIL was to solve the binary coding optimization problem, where P(xi) represents the probability that the variable takes a value of 1 at the ith gene position. However, the problem studied in this paper is multi-dimensional discrete resource scheduling problem. Therefore, integer encoding scheme is applied, that is, there are a variety of optional scheduling options in each position. Thus, our P(xi) in equation (6) denotes the probability value chosen for each alternative scheduling scheme at the ith gene position, which corresponds exactly to our problem here.

Secondly, the problem to be solved in this paper is an n-dimensional discrete problem, and the variables are independent of each other. Therefore, the probability of arbitrary solution can be expressed as:

$$P(X) = P(x_1, x_2, ..., x_n) = \prod_{i=1}^{n} P(x_i) \tag{7}$$

To calculate the probability of each component, the Monte Carlo method is used to represent the probability in terms of frequency. Let Sg(xi) be the set of values of the ith variable xi in the dominant population at the gth generation. The frequencies based on the statistics of the selected individuals can be denoted as F(xi = k), k=1,2,..,m. Note that, because there may be some variable xi = k that does not exist in the set Sg(xi). In view of this situation, in order to maintain a high population diversity of the population, we give a chance probability based on the average processing time of the cloudlet i and the overall average processing time. Thus, the improved marginal probability P(xi) can be expressed as equation (9).

$$P_g(x_i = k) = \begin{cases} F(x_i = k) \big/ N, & k \in S_g(x_i) \\ \beta \cdot \min(F(x_i = k) \big/ N), & otherwise \end{cases} \tag{8}$$

Where β =|A-avei| represents the chance coefficient, A denotes the average completion time of all cloudlets on all VMs, and avei is the average completion time of cloudlet i.

Next, a new more efficient adaptive learning rate function is proposed by referencing literature [7] to trade off exploration and exploitation of IEDA. The learning rate and the number of iterations are combined to form a natural and reasonable relation function (9).

$$\alpha = (\alpha_{up} - \alpha_{low}) * (g/G)^3 + \alpha_{low} \tag{9}$$

Here, αlow = 0.01 represents the lower bound of the learning rate, αup = 0.5 represents the upper bound of the learning rate, parameter g is the current generation, and G denotes the maximum number of iterations. The function expression shows that the initial growth rate of iteration is slow so that the proposed algorithm IEDA can capture a high capacity of exploitation. To the late iteration, in order to prevent the algorithm from falling into the local optimum, the learning rate need to be sharply increased to find the global optimal solution.

Then, as for population initialization, both the EMM and the Min-Min algorithm are introduced to obtain two initial chromosomes in the population initialization stage. In addition, we propose an IGA that generates partial initial individuals by repeating multiple crossover and mutation operations using the two chromosomes obtained by EMM and Min-Min, and then adds them to the population until the population size meets the termination criterion. The rest of initial individuals are generated at random. As for the concrete operation operator design of our proposed IGA, firstly, roulette method is chosen as the selection operator; secondly, the random single-point crossover and the parametric unified crossover are combined as the crossover operator, and finally, mutation operator is that two different gene positions are randomly selected for exchange.

Finally, our sampling strategy in IEDA is divided into two parts: one is probabilistic sampling by PBIL probability model; on the other hand, in order to increase the diversity of population, the crossover and mutation operation of IGA are applied to all individuals of this generation to generate new individuals. Furthermore, an improved and novel elitism criterion is employed, which converts the worst individual wg in this generation to the best individual bg-1 in the last generation after sampling.

Algorithm Presentation

After elaborating on the improvements of PBIL algorithm, our proposed IEDA is presented as Algorithm 1. Simultaneously, the population size of IEDA is set to N=n × m, and the maximum iteration G=5 × n is seen as a termination criterion.

TABLE. 1 After elaborating on the improvements

Algorithm 1: IEDA

```
Begin
1.  Set the initial population $Z_0$ be empty, selected population $S_0$ be empty, generation counter g=0.
2.  Initialization: Generate $Z_0 = \{X_1, X_2,…,X_N\}$ with the proposed population initialization method.
3.  For g=1 to $G$
        3.1 Evaluation: Calculate (5) for each solution $X$.
        3.2 Selection: $S_{g-1} \leftarrow$ Select the top $M$ individuals in term of objective values in non-decreasing order.
        3.3 Probabilistic Model
            For $i$ =1 to $n$
                Construct and solve (8);
            End For
        3.4 Sampling Strategy and Elitism Criterion
            $S_g(1) \leftarrow$ Sample individuals by probabilistic model (7);
            $S_g(2) \leftarrow$ Sample individuals by IGA;
            $S_g \leftarrow S_g(1)+ S_g(2)$ and apply elitism criterion: $w_g \leftarrow b_{g-1}$
    End For
4.  The best individual in $Z_G$ is the final solution.
End
```

EXPERIMENT ANALYSIS

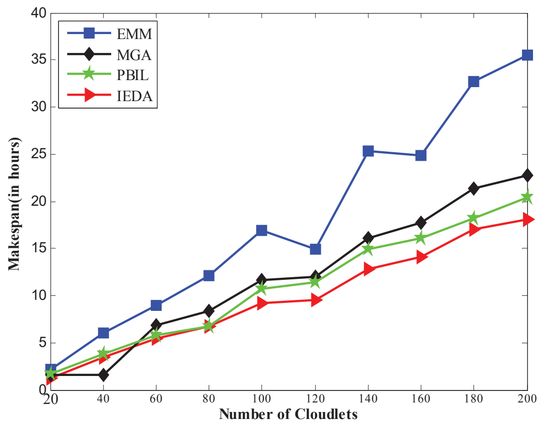
As mentioned earlier, there is no state-of-the-art algorithm for solving the studied problem, so three different comparative approaches: EMM[4], MGA [5], and PBIL[7] are chosen to compare with our proposed IEDA from two aspects: solution quality and convergence speed of all comparative algorithms. The CloudSim framework [9] was used to implement all comparative algorithms which are executed on the same conditions.

Solution Quality Comparisons

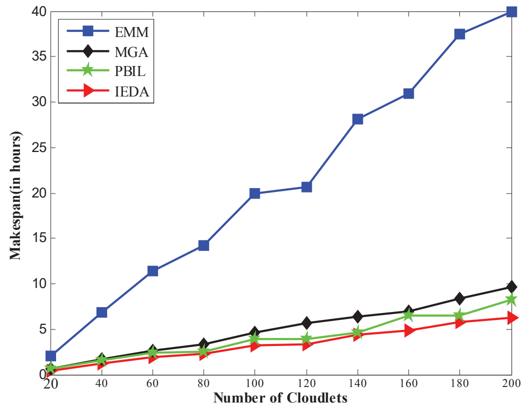
In this subsection, we will analyze the quality of the solution from two cases. Case1: the number of VMs is fixed, the number of cloudlets is ascending. We keep the number of VMs as 4 or 8, and increase the number of cloudlets from 20 to 200. Case2: the number of cloudlets is fixed, the number of VMs is ascending. We keep the number of cloudlet as 100 or 200, and increase the number of VMs from 4 to 12. The simulation data are randomly generated, in which the computing speed of the VMs and the size of the cloudlets come from the uniform distribution [100, 1000] and [1 × 105, 1 × 106], respectively.

Case 1: Fixed VMs and Varying Cloudlets

Fig.1 shows that our proposed IEDA has a better optimal solution whether the number of VMs is equal to 4 or 8. Meanwhile, the solution quality of PBIL is better than MGA, and EMM is the worst among all algorithms.



(a)

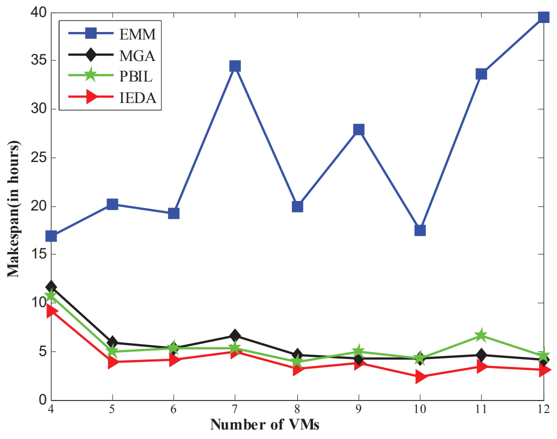


(b)

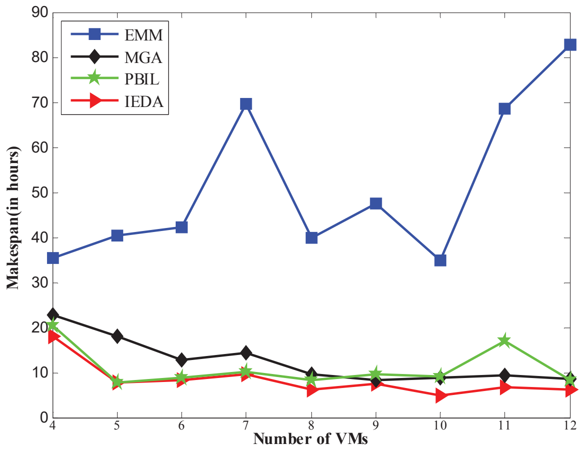
FIGURE 1. Results for fixed VMs and varying cloudlets on the configuration of (a) 4 VMs and (b) 8 VMs.

Case 2: Varying VMs and fixed Cloudlets

As seen from Fig.2, it is obvious that the proposed IEDA always outperforms other methods, no matter how the number of VMs changes. In addition, when the number of cloudlets equals 100, PBIL and MGA have the similar performance on solution quality, however when the number of cloudlets extends to 200, PBIL is basically better than MGA when the number of VMs is less than 9, and EMM is still the worst.



(a)



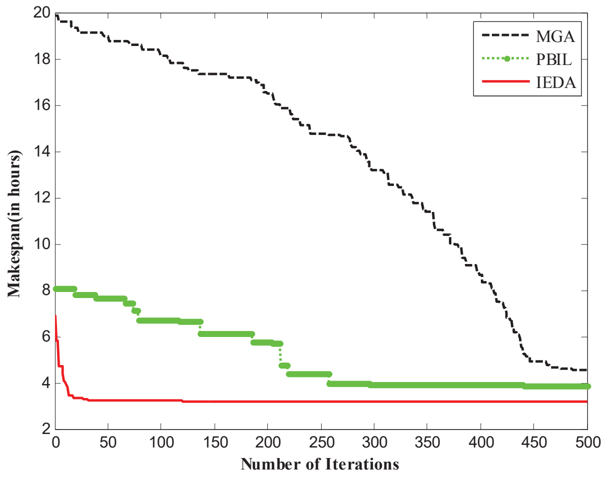
(b)

FIGURE 2. Results for fixed cloudlets and varying VMs on the configuration of (a) 100 cloudlets and (b) 200 cloudlets.

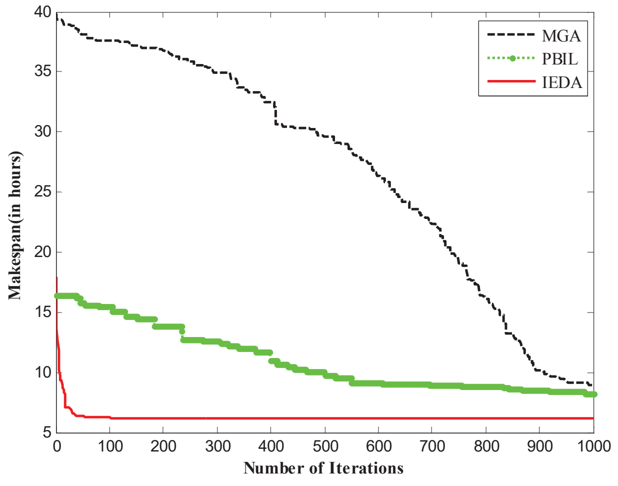
The Analysis of Convergence Speed

To analyze the convergence speed of MGA, PBIL and IEDA, we kept a record of the makespan of each generation on the configuration of keeping the number of VMs as 8, and the number of cloudlets as 100 or 200.

As we can see from Fig.3, the proposed IEDA has an absolute advantage in convergence speed among all



(a)



(b)

FIGURE 3. Results for convergence speed on the configuration of (a) 100 cloudlets and (b) 200 cloudlets.

Comparative algorithms. Note that, a better solution can be obtained from the early stage of evolution in our proposed IEDA since our method of generating initial population is different from others, which we combine two heuristic rules with stochastic methods to generate initial population sufficiently.

In conclusion, compared to other comparison algorithms, the proposed IEDA has achieved better results both in the solution quality and the convergence speed for tackling the problem studied in this paper.

CONCLUSION

In this paper, to schedule independent tasks in cloud computing, an IEDA is proposed to minimize makespan. The integer encoding scheme is used in this paper, and the method of probability calculation of PBIL algorithm is improved. Furthermore, to trade off exploration and exploitation of IEDA, an effective adaptive learning rate function is constructed. In addition, to generate the initial population reasonably, our proposed IEDA introduces two heuristics algorithm to generate two initial individuals which are evolved using the IGA to generate initial population. Finally, a hybrid sampling strategy is adopted in proposed IEDA. The experimental results show that our algorithm can give a better scheduling scheme than other comparative algorithms.

At present, cloud computing resource scheduling problem is still a hot issue. In this paper, we only consider single objective makespan and use the MIPS to measure VM performance without considering other parameters such as bandwidth. In addition, EDA has many other optional probabilistic models, either variable-dependent or variable-independent. Interested researchers can try other probabilistic models to solve more complex problems. There is still vast space for further research in this area.

ACKNOWLEDGMENTS

This work was support in part by the National Natural Science Foundation of China (No.71671168), the China Postdoctoral Science Foundation (No.2015M581707), the Jiangsu Planned Projects for Post-doctoral Research Funds (No.1501006B).

REFERENCES

1. Z.-H. Zhan, X.-F. Liu, Y.-J. Gong *et al.*, "Cloud Computing Resource Scheduling and a Survey of Its Evolutionary Approaches," *ACM Computing Surveys,* vol. 47, no. 4, pp. 1-33, Jul. 2015.
2. X. L. Liang, H. P. Chen, and J. A. Lozano, "A Boltzmann-Based Estimation of Distribution Algorithm for a General Resource Scheduling Model," *Ieee Transactions on Evolutionary Computation,* vol. 19, no. 6, pp. 793-806, Dec. 2015.
3. P. Kumar, and A. Verma, "Independent task scheduling in cloud computing by improved genetic algorithm," *International Journal of Advanced Research in Computer Science and Software Engineering,* vol. 2, no. 5, May. 2012.
4. U. Bhoi, and P. N. Ramanuj, "Enhanced max-min task scheduling algorithm in cloud computing," *International Journal of Application or Innovation in Engineering and Management (IJAIEM),* vol. 2, no. 4, pp. 259-264, 2013.
5. S. Singh, and M. Kalra, "Scheduling of independent tasks in cloud computing using modified genetic algorithm," *in 2014 International Conference on Computational Intelligence and Communication Networks (CICN)*, 2014, pp. 565-569.
6. P. Larranaga, and J. A. Lozano, Estimation of distribution algorithms: A new tool for evolutionary computation: Springer Science & Business Media, 2002.
7. N. Chen, X. Fang, and X. Wang, "A cloud computing resource scheduling scheme based on estimation of distribution algorithm," *in Proceedings of 2nd International Conference on Systems and Informatics (ICSAI 2014)*, Nov. 2014, pp. 304-308.
8. S. Baluja, Population-based incremental learning. a method for integrating genetic search based function optimization and competitive learning, *Carnegie-Mellon Univ Pittsburgh Pa Dept Of Computer Science*, 1994.
9. R. N. Calheiros, R. Ranjan, A. Beloglazov *et al.*, "CloudSim: a toolkit for modeling and simulation of cloud computing environments and evaluation of resource provisioning algorithms," *Software: Practice and Experience,* vol. 41, no. 1, pp. 23-50, Jan. 2011.