An Adaptive Immune Genetic Algorithm for Machinery Manufacturing Task Scheduling in Cloud Computing System

Jie Zhang

Dongguan University of Technology, Dongguan 523808, China
Corresponding author: zj0769@126.com

Corresponding author is Jie Zhang

Abstract
In recent years, cloud computing has been focused as a new mode of service in the field of computer science. Cloud computing aims to maximize the benefit of distributed resources and aggregate them to achieve higher throughput to solve large scale computation problems. A new manufacturing paradigm, i.e. Cloud Manufacturing (CMfg), which uses cloud computing to solve task scheduling problem, has been proposed recently. In order to satisfy high efficiency and low cost machinery design task scheduling in CMfg, a new energy adaptive immune genetic algorithm (EAIGA) was proposed. Combining potential energy storage and detection, the new algorithm can not only improve searching diversity based on immune strategy, but also adaptively adjust the probabilities of crossover and mutation with low time complexity. The experimental results show that the new algorithm can effectively solving the manufacturing task scheduling problem with a good balance between searching diversification and intensification.
Keywords: ADAPTIVE IMMUNE GENETIC ALGORITHM, CLOUD COMPUTING, MANUFACTURING TASK SCHEDULING, LOW TIME COMPLEXITY.

1. Introduction

Cloud computing is an emerging paradigm that accesses network and shares computing resources with convenient and minimal management efforts. It is one of the smart technologies that will reshape the world and shifts Information Technology infrastructure to third party to be available to the customers as commodities [1]. The computing environment of cloud computing can be outsourced to another party to use the computing power or resources via Internet. Emerging of this technology moves the computing power and data from personal computer and portable devices into large data centers. End-users access and use all the services without knowing the physical location and the configuration of the system at the providers' sides [2].

Cloud computing platform guarantees subscribers that it sticks to the service level agreement (SLA) by providing resources as service and by needs [3]. It allows business customers to scale up and down their resource usage based on needs. Resources are shared in the cloud computing environment; if they are not properly distributed then the result will be resource wastage. In Infrastructure-as-a-Service (IaaS) clouds, dynamic resource allocation is exploiting by the parallel data processing frame work. This system fits well in a cloud for efficiently parallelizing incoming set of tasks using large data [4]. Efficiency of resource usage and dynamic resource provisioning capabilities of the VM are improved by the help of VM related features such as flexible resource provisioning and migration.

Based on the cloud computing technology, a new manufacturing paradigm named Cloud Manufacturing (CMfg) [5] was presented by the authors' group for the application of information technologies in manufacturing. "Design as a Service" (DaaS) in collaborative design is an important part of CMfg. It not only encapsulates multi-disciplinary knowledge, design group, processing hardware and software into cloud services, but also provides design programming and task scheduling strategies as services [6]. For providing agile and intelligent strategy services on demand, high efficient task scheduling algorithms are imperative in CMfg.

In machinery manufacturing, it is in the distributed design phases where each designer or team works on specific subtasks, that managing task interdependencies becomes more crucial. It is also emphasized by work at distance as compared to co-location (such as in open-plan offices). It involves aligning work via various co-ordination mechanisms. Machinery manufacturing task scheduling means reorganizing and scheduling tasks of product design among execution units (i.e. clusters, simulation resources and human and so on) at the initial stage of development [7]. It's a kind of programming which handles the relationship between tasks and design units and makes sure parallel execution of design subtasks in resources. In CMfg, it ensures efficient utilization of resources and was provided as cloud services for distributed collaborative production. Users can easily participate in collaboration and finish their tasks without buying any expensive devices.

Machinery manufacturing task scheduling has been proved as an NP-hard problem [8]. That is to say, it's hard to be solved in polynomial time by using precise methods. Thus, researches in scheduling are most likely to focus on the improvement and application of intelligent optimization algorithms [9] in solving this problem. Genetic algorithm (GA) [10] which is one of the most classical intelligent optimization algorithms was inspired by the principle of evolution and natural genetics. For satisfying the high efficiency and fast response requirements of scheduling services in CMfg, this paper presents a new improved energy adaptive immune genetic algorithm. Combining with the strong intensification ability of immune strategy, the idea of energy conversation and storage is introduced to GA for improving diversification. Compared with niched strategy, it is very simple with low complexity, and it shows high performances in simulation results on standard scheduling tests.

The rest of the paper is organized as follows. Section 2 describes the general scheduling model and cloud computing system for collaborative manufacturing task. Section 3 introduces the immune genetic algorithm (IGA) and then elaborates the new improved energy adaptive immune genetic algorithm (EA_IGA). Section 4 gives the experimental results and discussions of the new improved algorithm in solving scheduling problems. Then conclusion is drawn in Section 5.

2. Manufacturing Task Scheduling Model and Cloud Computing System

2.1. Manufacturing Task Scheduling Model

Collaborative manufacturing task scheduling problem (CMTSP) focuses only on execution time of each task without considering any communication factors. However, relationship among designing tasks are usually complex and the QoS (Quality of Services) constraints of each design unit are more. In general, it is assumed that a collaborative project is decomposed into a group of design tasks and each task is inseparable. Tasks are ordered and parallelizable and can only be executed in one unit during a certain time.

In CMfg, design units can be human, organization, devices or software. Different design unit has different execution efficiency. The execution time of task is determined by design units' execution capability, preference, work environment and reliability, and so on. This paper primarily focuses on scheduling algorithms, so it is assumed that these execution times of tasks in different design units have been given without change and represented as a time vector. Then CMTSP can be defined as follows.

Let $T=\{t_1,t_2,...,t_n\}$ be a set of manufacturing tasks, and n be the number of tasks. $H=\{h_1,h_2,...,h_m\}$ denotes a set of design unit services, and m is the number of services. Then let $P(i)$ be the predecessor task set of task t_i, and $S(i)$ be its successor task set. Tasks are non-preemptive and each task can only be started after all its predecessor tasks are finished. Considering the high-priority tasks in the same design units, if h_j was selected for t_i, let $hp_j(i)$ be these high-priority task set. Then

$$startTime(t_i)=\max_{x\in P(i)\cup pr_j(i)}\left(endTime(x)\right) \tag{1}$$

To simplify this problem, it is assumed that the 'units–tasks' execution time vector as follow:

$$exeTime_{m\times n}=\left\{t_{ij}\mid 1\le i\le m,1\le j\le n\right\} \tag{2}$$

Then the objective function can be represented as:

$$f=\min\left(exeTime(T)\right)=\min\left(endTime(t_n)\right) \tag{3}$$

2.2. Cloud Computing System

Cloud computing is the development of computing and grid computing, parallel, is a kind of distributed computing, its basic idea is through the network will be huge computing program automatically split into numerous smaller subroutine, to pay the huge system consists of multiple servers, calculation and analysis after the treatment results back to the user through a searcher, and the "cloud" exists to provide these resources of the network. Cloud computing provides service for users is huge, so the "cloud" in the task is enormous.

No matter what the scholars defines cloud and cloud computing, in all the definitions, the user can get better service through the browser, desktop applications or mobile applications to access cloud services. To promote the view that cloud computing allows companies to deploy the application more quickly, and reduce the complexity and cost of maintenance management, and allows the IT resources rapidly redistributed to cope with rapid change of enterprise demand. Fig. 1 shows the cloud computing system.

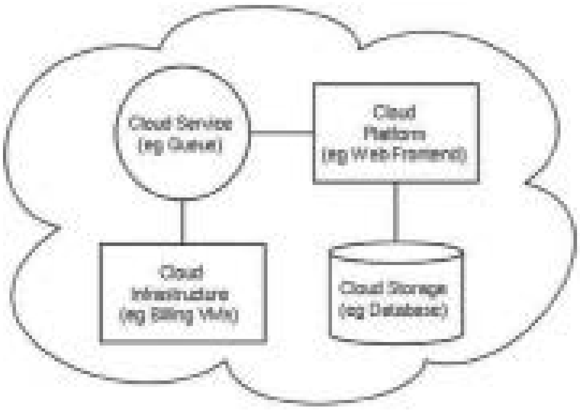

Figure 1. The Cloud Computing System for Manufacturing Task Scheduling

3. A New Adaptive Immune Genetic Algorithm

Based on the above mentioned model, this paper aims at solving CMTSP with high efficiency and low time consuming. As mentioned in Section 1, immune strategy introduces heuristic information into algorithm and uses the concepts of antigen recognition and immune regulation to enhance GA's performance in terms of intensification. It is simpler than local search strategy (local traversal), but is easy to trapped into local convergence if the heuristic information are inaccurate. Therefore, based on IGA, this paper presented a new strategy to improve algorithm diversification and balance the strong intensification of immune strategy.

IGA contains genetic evolution and immune vaccination. But first of all, antigen extract and vaccine selection according to the feature information of problem is the most important part of this algorithm. It is a core rule to lead the population evolution into the right direction. Then, the population initialization and the genetic evolution are all the same as the standard genetic algorithm. After selection, crossover and mutation, new populations are vaccinated by antibodies. If individual in the new populations are better than its parent, then accept it. Or the individual is accepted in probabilistic according to the rules of simulated annealing.

According to the principles of energy storage and release in nature, energy in object is transformed between kinetic and potential without loss. It is supposed that the individual degeneration be the process of potential energy release and the individual evolution be the process of potential energy storage. The potential energy can clearly reflect the dynamic change of individuals in each generation. The diversity strategy can then be applied in time according to it.

To be more specific, it is assumed a population be $PL = \{pl_i \mid 1 \le i \le N\}$, where N is the number of individuals. An individual pl_i is composed by gene bits, fitness value (fitness) and potential energy (potential) in algorithms. The potential energy represents the fitness improvement of individual in each generation, that is:

$$pl_i(t)_{potential} = pl_i(t)_{fitness} - pl_i(t-1)_{fitness} \tag{4}$$

where t represents the evolution iteration.

Based these definition, a new operation called "Potential Detection" was added into IGA and the original crossover and mutation operations are then changed into adaptive. The specific process of the new improved algorithm is given below.

First, to dynamically escape the local convergence in IGA, potential detection operation is applied before selection. In potential detection, the potential energy of each individual is calculated according to Eq. (4). If the potential energy is negative, it indicates that the individual is degenerated in this generation. Then the individual will be regenerated with probability X_p and here is defined as

$$X_p = \exp\left(-A \times pl_i(t)_{fitness} / pl_{best}(t)_{fitness}\right) \tag{5}$$

In the above equation, pl_{best} represents the best individual in population and A is a control parameter. The pseudo code of potential detection operation is shown as follow.

$Potential\_Detection\ (PL)$
for $i = 1 : N$
  $if\ pl_i(t)_{potential} < 0$
    $X_p = \exp\left(-A \times pl_i(t)_{fitness} / pl_{best}(t)_{fitness}\right)$
    $Randomly\ generate\ 0 < \lambda < 1$
    $if\ p > X_p$
      $Regenerate\ a\ new\ pl_{tmp}(t)$
      $pl_i(t) = pl_{tmp}(t)$
      $pl_i(t)_{potential} = 0$
    end if
  end if
end for

In this operation, X_p changes in every generation. The worse the individual's fitness value, the less X_p is, and then the higher probability of individual regeneration is. Therefore, individuals can be changed according to their own situation. Its time complexity is $O(N)$, which is far lower than niched strategy.

To further balance the diversification and intensification in IGA, a new parameter adaptive strategy is introduced into crossover and mutation based on individual's potential energy. Let X_{cp} and X_{mp} be the crossover probability and the mutation probability respectively. Then X_{cp} and X_{mp} are adaptively changed according to each individual's potential energy in every generation. The pseudo codes of the specific adaptive strategy are listed as follow.

for $i = 1 : N$
  $X_{cp} = 1/\left(1 + \exp\left(B \times pl_i(t)_{potential}\right)\right)$
  $X_{mp} = 1/\left(1 + \exp\left(C \times pl_i(t)_{potential}\right)\right)$
  $if\ X_{cp} < 0.5$
    $X_{cp} = 0.5$
  else if $X_{cp} > 1$
    $X_{cp} = 1$
  end if
  if $X_{mp} > 0.5$
    $X_{mp} = 0.5$
  end if
  Perform crossover operation in probability X_{cp}
  Perform mutation operation in probability X_{mp}
end for

Here, B and C are control parameters of the adaptive strategy. In the new adaptive strategy, $X_{cp} \in [0.5, 1]$ and $X_{mp} \in [0, 0.5]$. When the individual's potential energy is smaller, X_{cp} and X_{mp} become larger to improve the global searching ability of algorithm. If the individual's potential energy is larger, then X_{cp} and X_{mp} become smaller to adapt the local search in algorithm. Compared with the traditional adaptive method, it removes the calculation of individuals' average fitness of population in iteration, so the time complexity is a bit lower.

4. Experiments and Discussions

Two kinds of directed acyclic graph (DAG) are used to study the efficiency and feasibility of PAIGA in the experiments. The DAG includes 30 tasks with 4 units and 60 tasks with 4 units which are selected from the standard PSPLIB for the computation tests. The execution times of each algorithm are randomly generated in the range [1, 10]. And the time will be stored in a document with text formatting after the calculation. Then the heuristic information of CMT-SP can be defined as follows.

$$Heualgthm = \min_{j \in [1,n]} (exeTime(hp_j(i) + t_{ij})) \tag{6}$$

The roulette wheel selection algorithm of genetic algorithm is commonly used to select the individual with the maximize fitness value with higher probability. Since the traditional GA is easily trapped in the local optimum and appeared premature convergence. This method can avoid the individuals trapping in the local optimum. And then the objective function is set as:

$$\max\ function = \frac{M}{exeTime(T)} \tag{7}$$

In the equation, M is a constant which makes the object fitness value in the algorithms not too large or too small. Considering the results are easy for us to observe, the parameter M is set as 500 in these experiments.

Then the comparison experiments are designed as follows. The four kinds of GA are taken into consideration which are standard GA, IGA, PIGA (IGA with only potential detection)), EAIGA (IGA with both potential detection and adaptive crossover and mutation). In order to guarantee the comparability of the results, the execution parameters are set in the following ways. In the GA and IGA, the Crossover probability is 0.75 and the mutation probability is set 0.12. In potential detection and adaptive strategy, A=5, B=1, C=2. Considering the execution time and equity of these algorithms, the population sizes of all the tested algorithms are set to 30 and the maximum number of generation is set to be 1000. So the iteration calculation of the program is limited to 1000. And 80 runs of each experimental setting are designed and the average fitness value of the best solutions in each algorithm throughout the run is recorded.

The experiments results of the total runs are shown in the Fig. 2 and Fig. 3. It is obvious for us to find the average best fitness value of the EAIGA with both potential detection and adaptive crossover and mutation is the bigger than these of the other three algorithms after 80 runs in the experiments. The improvements from GA to EAIGA are step by step. Based on the two figures, the evolution trends in the situation of both 30 tasks and 60 tasks are the same. It is easy for us to find the best fitness value is bigger and bigger with the change of the iteration times. In this way, the calculation of each algorithm is trapped in the local optimum and appeared premature convergence in the early stage of the iterations. Based on GA, IGA improved GA by the introduction of heuristics (vaccination). But the trends also indicated that heuristics may lead the algorithm trapping into local optimization in early generation. According to the curves of IGA in Fig. 1, it is clear that algorithm is always trapped into local convergence in about 100 iterations. With the increase of the iteration times, the best fitness values become bigger which means the calculation jumps out the local optimum. But after applying potential detection and parameter adaptation in algorithm, it can be seen that the improvements in two steps are remarkable and the best fitness value of the EAIGA is the biggest with the same iterations in each algorithm. The calculation results of the EAIGA are the best solution for the optimization problem in this research. And according to the optimization results, the method EAIGA is better than the other three kinds of genetic algorithms for solving this kind problem proposed in this research.

The execution times for the four genetic algorithms are illustrated in the Fig. 4. Compared with these results, the efficiency of each algorithm can be easy to find. It is easy for us to calculate the points of the worst solution, the best solution, the average solution, standard deviation and time consuming in 80 runs based on the figures above. The performance of the searching ability for the IGA is better than the standard GA according to Fig. 4. But the dynamic calculation of heuristic in each generation is very time consuming. What's more, from the standard deviation, we can found out that IGA is more unstable than GA. After applying potential detection operation, the searching ability was improved with a little more time consuming. But the value of standard deviation still increased and algorithm became more unstable. For solving this problem and balance intensification and diversification, parameter adaption based on individual's potential energy was introduced. It can be seen that EAIGA is more stable than IGA and PIGA

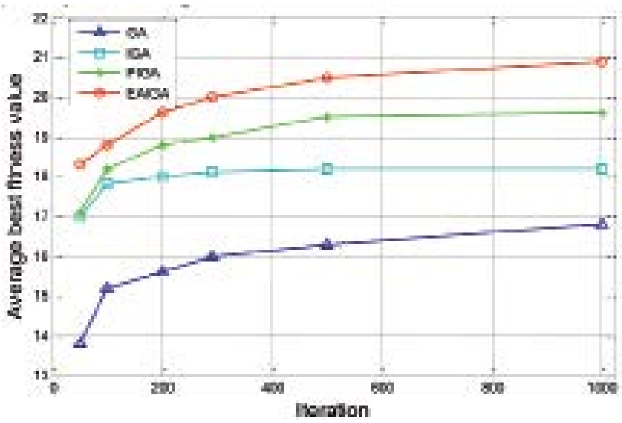

Figure 2. The Average Best Fitness Values of the Four GA Algorithm in the 30 tasks with 4 units

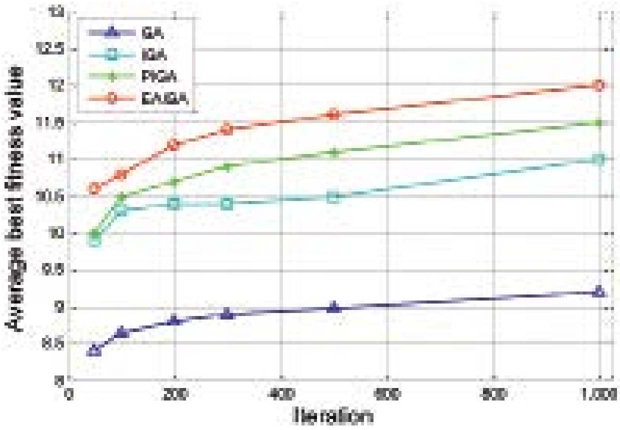

Figure 3. The Average Best Fitness Values of the Four GA Algorithm in the 60 tasks with 4 units

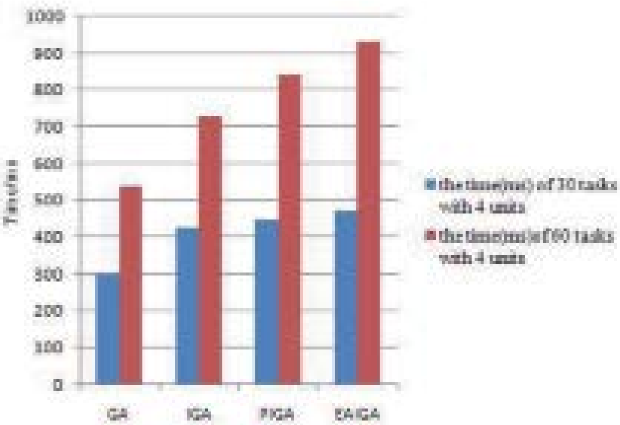

Figure 4. The Execution Times in the Two Kinds of the Directed Acyclic Graph (DAG) in the Experiments

with lower standard deviations. Although time consuming was increased a little, the solution quality was improved more.

Therefore, from the global perspective, EAIGA showed high performance in both 30 tasks scale and 60 tasks scale. Potential detection had improved the algorithm's diversity and parameter adaptation had improved the algorithm's stability. Both two strategies cost not much time and increased the solution quality. Without the increase of time complexity based on IGA, EAIGA showed quite good balanced capacity and searching ability for addressing task scheduling problems.

5. Conclusion

A new improved energy adaptive immune genetic algorithm for addressing CMTSP in cloud computing system is presented in this research. It is inspired by energy conversation and storage. And the EAIGA has better balanced capacity and searching ability for addressing task scheduling problems than the similarity genetic algorithm. Based on immune strategy, it can diversify the population in generation and increase the stability of algorithms. Most importantly, big improvement in solution quality is achieved. From the perspective of solution quality and stability, it obtains big improvement and satisfies the requirements of high efficiency to some degree in providing scheduling strategy services in CMfg.

However, on the other hand, the energy strategy doesn't reduce the total time consuming of IGA for addressing CMTSP and still couldn't satisfy the requirement of agility in a way. So, future research needs to focus on how to decrease the time complexity of immune strategy and decrease the total time consuming of EADIGA for more efficient scheduling in CMfg.

Acknowledgement

This work is supported by the Scinece and Technology Project of Colleges and Institutes of Science Research of Dongguan (No.2012108102010).

References

1. Ghanbari, S. and Othman. M. A Priority Based Job Scheduling Algorithm in Cloud Computing. Procedia Engineering, 2012, vol. 50, pp. 778-785.
2. Xu, B.M., Zhao, C.Y., Hu, E.Z. and Hu, B. Job Scheduling Algorithm Based on Berger Model in Cloud Environment, Advances in Engineering Software, 2011, vol.42, pp.419-425.
3. Palmieri, F., Buonanno, L., Venticinque, S., Aversa, R. and Di Martino, B. A Distributed Scheduling Framework Based on Selfish Autonomous Agents for Federated Cloud Environments, Future Generation Computer Systems, 2013, vol.29, pp.1461-1472.
4. Mezmaz, M., Melab, N., Kessaci, Y., Lee, Y.C., Talbi, E.G., Zomaya, A.Y. and Tuyttens, D. A Parallel Bi-Objective Hybrid Metaheuristic for Energy-Aware Scheduling for Cloud Computing Systems, Journal of Parallel and Distributed Computing, 2011, vol.71, pp.1497-1508.
5. Tsai, J.T., Fang, J.C. and Chou, J.H. Optimized Task Scheduling and Resource Allocation on Cloud Computing Environment Using Improved Differential Evolution Algorithm, Computers & Operations Research, 2013, vol.40, pp.3045-3055.
6. B H Li, L Zhang, S L Wang, F Tao, J W Cao, X D Jiang, X Song, X D Chai. Cloud manufacturing: a new service-oriented networked manufacturing model. Computer Integrated Manufacturing Systems, 2010, vol.16, pp.1-16.
7. L Zhang, Y. L. Luo, F Tao, L Ren, H Guo. Key-technologies for the construction of manufacturing cloud, Computer Integrated Manufacturing Systems, 2010, vol.16, pp.2510-2520.
8. J. D. Ullman. NP-complete scheduling problems, Journal of Computer and System Sciences, 2005, vol.10, pp.384-393.
9. G F Yin, Y Luo, H N Long, E J Cheng. Genetic algorithms for subtask scheduling in concurrent design. Journal of computer-aided design & computer graphics, 2004, vol.16, pp.1122-1126.
10. J B Zhang, K Cheng. An adaptive ant colony algorithm for concurrent design task planning problem, Journal of computer-aided design & computer graphics, 2010, vol. 22, pp.1070-1074.