Future Computing and Informatics Journal

Volume 3
Issue 2 *Vol 3,Issue 2,2018*

Article 18

2018

A genetic algorithm for service flow management with budget constraint in heterogeneous computing

Ahmed A. AbdulHamed
Department of Computer Science, Menoufia University, Egypt, ahmed_abd1989@yahoo.com

Medhat A. Tawfeek
Department of Computer Science, Menoufia University, Egypt, medhattaw@yahoo.com

Arabi E. Keshk
Department of Computer Science, Menoufia University, Egypt, arabikeshk@yahoo.com

Available online at www.sciencedirect.com

ScienceDirect

Future Computing and Informatics Journal 3 (2018) 341−347
http://www.journals.elsevier.com/future-computing-and-informatics-journal/

A genetic algorithm for service flow management with budget constraint in heterogeneous computing

Ahmed A. AbdulHamed*, Medhat A. Tawfeek, Arabi E. Keshk

Department of Computer Science, Menoufia University, Egypt

Received 1 April 2018; accepted 18 October 2018
Available online 30 October 2018

Abstract

Heterogeneous computing supply various and scalable resources for many applications requirements. Its structure is based on interconnecting machines with several processing capacity spread over networks. The scientific bioinformatics and many other applications demand service flow processing in which services have dependencies execution. The environments of this computing are suitable for huge computational needs that contains diverse groups of services. Managing and mapping services of service flow to the suitable candidates who provides the service is classified as NP-complete problem. The managing such interdependent services on heterogeneous environments also takes the Quality of Service (QoS) requirements from users into account. This paper firstly proposes a model of service flow management with service cost quality requirement in heterogeneous computing. After that a service flow mapping algorithm named genetic to reduce the consumed cost of an application in heterogeneous environments is proposed. This algorithm gives a robust search technique that allow a soft cost solution to be derived from a huge search space of solutions by inheriting the evolution concepts. The obtained results from the applied experiments prove that genetic can save more than fifteen percent from the cost and also outperforms the compared algorithms in the metric of speedup and SLR.

Keywords: Heterogeneous computing; Service flow; Genetic algorithm; Service cost

1. Introduction

Heterogeneous computing proposes completely diversified computing nodes that have different capabilities with various ways for instructions execution. The advantages of getting diversified kinds of computing nodes are the performance up and energy Effectiveness [1]. Diversity challenges exist at the hardware level and software level. The most two common within these challenges are scalability and distributing the incoming workload among the various candidate to induce the towering performance [2]. Heterogeneous computing may be consider as a service supported model that capable of supporting various computing services network. Scientific service flows usually need different resources to manage computation activities of large data. A service flow management system is used for managing these applications by hiding execution details on resources provided by heterogeneous service candidates [3]. So as to constrict the price of service flow, effective strategies by relying on meta-heuristics are needed for mapping and managing the services [4]. In this paper a model for service flow management is proposed. It contains four separated modules. The management module from these modules is considered as the brain module of the proposed model. Also a genetic algorithm for service flow mapping based on budget QoS constraint is proposed.

Genetic algorithm (GA) is employed as a meta-heuristic technique that's supported natural evolution. GA combines the exploration and exploitation ideas. Exploration discovers new solves from solution area. Exploitation exploits the most effective

* Corresponding author.
E-mail addresses: ahmed_abd1989@yahoo.com (A.A. AbdulHamed), medhattaw@yahoo.com (M.A. Tawfeek), arabikeshk@yahoo.com (A.E. Keshk).

Peer review under responsibility of Faculty of Computers and Information Technology, Future University in Egypt

https://doi.org/10.1016/j.fcij.2018.10.004

solutions from previous searches. The solution is represented by chromosomes [5]. Chromosomes may be described by bit strings or symbolic expressions depending on current application. The looking for a suitable chromosomes begins with a population of initial chromosomes. Current population members help in provoking new population using selection, mutation and crossover operation that simulate biological evolution [6]. Fitness function employed to measure each chromosome quality at each generation. GA has been implemented to a variety of optimization such as robot control and results in a great success [7]. The rest of this paper is arranged as follows. Section 2 scans the proposed service flow management model and the service flow mapping problem. Section 3 provides the most important related work and overviews the standard genetic algorithm. Section 4 shows the details of the proposed genetic for service flow management. In, section 5 discuss the performance of the proposed genetic with various experiments. Section 6 finally, gives a summarization and some future trends.

2. Heterogeneous service flow model

This section proposes the heterogeneous service flow management model. The model consist of four modules: service repository module, management module, broker module and Service Level Monitoring (SLM) module.

1. Service repository contains all kinds of heterogeneous services.
2. Management module maintains a management algorithms to generate best mapping according to user's QoS requirement.
3. Broker module collects QoS requirement and all suitable candidates of needed services from repository and transforms these information to management module.
4. SLM module is to monitor the execution stages of services and feedback the monitoring results to broker module.

The proposed service flow management model for heterogeneous environment is shown in Fig. 1.

- User submits a separate services or service flow with QoS requirement specification based SLM to the broker module.
- Broker module copy services needed for the separate services or service flow from repository, and then send the separate services or service flow and the list of suitable services candidates to management module.
- Broker module also negotiates and configures service for user according to SLM between them.
- Management module generates a best mapping according to current scenario. After that it transmits the services to suitable candidate according to the reach mapping and acknowledges the broker.
- SLM module monitors and tracks the execution and sends the feedback from tracking results to broker module.

The service in this paper is modelled as *S* and *Scost* stand for service cost. *SN* is the number of services in Service repository. The user required service is composited of *N* services, the flow of service is modelled as *FS* that is represented by a directed acyclic graph (DAG) $FS = (V, A)$, where $V = \{S_1, S_2, \ldots S_N\}$ corresponds to the needed service requirement of the *FS*. The set of arcs *A* represents precedence relations between services. The Fig. 2 presents an example of *FS* that contains seven services.

Every service S_i in *FS* has a selection domain that represents the service candidates $SC = \{C1, C_2 \ldots C_m\}$ such that *SC* represents the set of suitable service candidates and *m* is the number of *SC* candidates. The main goal from Service flow management module is to find a best mapping to *FS* that satisfies the QoS and optimizes the objective that user specified.

In this paper our concentration is based only on the cost constraint. The cost of service flow *FS.cost* should be not more than the user specified budget in SLM. The *FS.cost* is computed by Eq. (1).

$$FS.\cos t = \sum_{i=1}^{N} S_j^i.\cos t \le Budget \qquad (1)$$

where S_j^i is the service Si that mapped to candidate C_j. It is seen from Eq. (1) that for cost optimization, the goal of the managing algorithm is to find a map that satisfies the cost constraint or minimizes the value of *FS.cost*.

3. Related work and standard genetic algorithm (SGA) overview

3.1. Related work

There are a lot of researches for heterogeneous computing scheduling related to task and workflow scheduling but there are less researches for service flow management. This subsection scans the most popular related work for scheduling in heterogeneous computing. Min−Min in Ref. [3] and Max-Min in Ref. [8] are used to satisfy various constrains of QoS such as time and/or cost. The research work in Refs. [4,9] handles the problem of scheduling the tasks in sections with sundry tasks using branch model likes Markov decision and depends on the iteration method of cost objective. Genetic based optimization techniques also have been used to tackle grid scheduling problem as in Refs. [7,10,11]. Although these approaches worked effectively in grid environment, they couldn't be directly applied to solve scheduling problem in heterogeneous computing. The workflow based particle swarm optimization (PSO) focusing on cost reduction of application execution is proposed in Ref. [12]. ACO in Ref. [13] is used to solve workflow with diverse QoS needs for grid computing. The research work in Ref. [14] introduced a Multiple QoS constrains scheduling strategy of multi-workflows in cloud computing to tackle workflow scheduling problem. ACO, GA, and PSO in Ref. [15] are used to tackle cloud workflow management. The obtained results show that the performance of ACO outperformed PSO and GA methods. The research work in Ref. [2] proposes a novel Greedy- Ant workflow management algorithm to

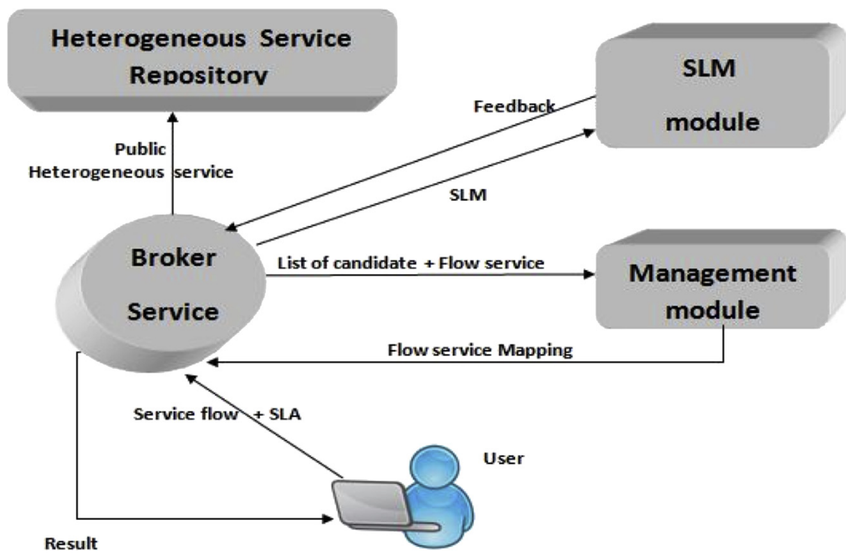

Fig. 1. The proposed service flow model.

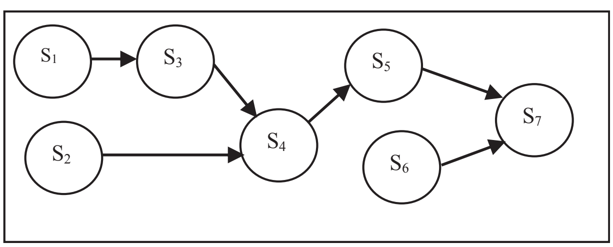

Fig. 2. Service flow example Flowchart.

minimize total execution time of an application in heterogeneous environments.

3.2. Standard genetic algorithm (SGA) overview

SGA is an efficient search method motivated by biological evolution. SGA generates successor chromosome by iteratively recombining and mutating parts of the best currently chromosome. The algorithm operates by repeatedly updating a pool of chromosome, called the population. All chromosomes from each population are ranked by a given fitness function each iteration. The fitness value shows the quality of chromosome compared to the others chromosomes in the population [16]. A new generation is then fabricated by selecting specific chromosomes from the existing population. Some of these selected chromosomes are copied into the next generation population and the others are used in crossover and mutation operation to create new offspring chromosomes [5]. The flowchart of SGA is presented in Fig. 3.

A typical SGA includes the following steps [5] [6]:

1. Creating initial population of chromosomes randomly.
2. Estimating the fitness value of each chromosomes in the current population and save the best one.
3. Generating new offspring by applying genetic operators (selection, crossover and mutation).
4. Repeating steps 2 and 3 until the algorithm terminates.

4. Proposed genetic algorithm for service flow

Service flow management focuses on mapping and managing the execution of dependent services on diverse candidates. In order to using genetic algorithm concept to solve the service flow mapping problem, the chromosome representation in the population, the fitness function and genetic operations should be determined. The details are presented in following subsections.

4.1. Chromosome representation

Each chromosome in the population represents a feasible solution to the problem, and consists of a vector of suitable candidates for services assignments. Fig. 4 shows the chromosome representation. It is shown from Fig. 4 that we have a set of candidates for each service and the chromosome is represented as a vector of selected candidates. The vector length equals *N* that represent the number of services in *FS* as shown in the bottom of Fig. 4.

4.2. Fitness function and selection

A fitness function is used to measure the quality of each chromosome in the current generation. As the goal of the proposed genetic is to minimize the total cost of FS execution. After calculating the fitness for each chromosome, the selection operation is applied. The proposed genetic algorithm depends on the roulette wheel selection method that is works as follows. The chromosomes are packed into circle of contiguous segments, such that each segment is proportional to chromosome fitness. A random number is generated and the chromosome whose segment spans the generated number is selected. This process is repeated until the specific number of chromosome is selected.

4.3. Genetic operators

Genetic operations comprise chromosomes in the existing population and generate new ones. Two genetic operators:

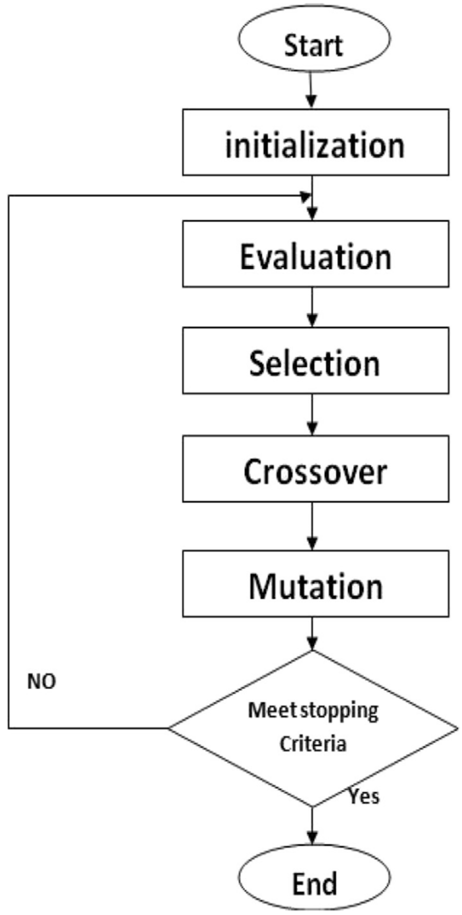

Fig. 3. Standard genetic algorithm (SGA) flowchart.

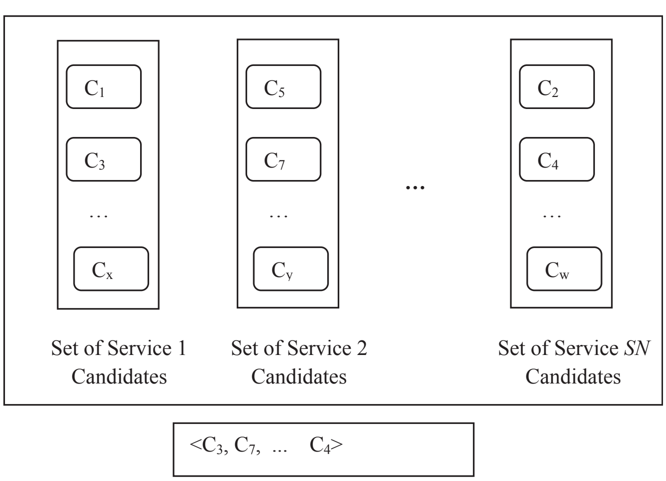

Fig. 4. Chromosome representation.

crossover and mutation is implemented for the *FS* mapping problems. Crossovers are used to generate new chromosomes from the current generation by combining specific parts of the selected chromosomes. The main target from crossover is that it may produce better chromosomes with high quality. The proposed algorithm based genetic utilizes the order-based crossover method that works as follows. Several genes are selected randomly from one chromosome (parent1) and then the order of those genes is imposed on the respective genes in the other chromosome (parent2). For example:

Chromosome (Parent1): 7 **5** **0** 3 **6** 1 4 2

The letter C is removed from the chromosome representation for simplicity. The genes in bold are the randomly selected genes. Now, the order 5, 0, then 6 is applied on the same genes in Parent2 to give Offspring chromosome as following:

Chromosome (Parent2): 4 3 6 7 2 5 1 0

Offspring1: 4 3 5 7 2 0 1 6

The same steps are applied between these two parents conversely to get the second offspring.

In the proposed genetic algorithm, mutations are used to o explore a new solution. It allows a certain offspring to obtain new features that are not in its parents. The proposed genetic algorithm depends on insertion mutation method. It is a very efficient approach for mutation that operates as follows. Only one gene (ie 3) is chosen to be displaced and inserted back into the same chromosome as following:

1 2 3 4 5 6 0 7
Take the 3 out of the sequence,
1 2 4 5 6 0 7
and reinsert the 3 at a randomly chosen position:
1 2 4 5 3 6 0 7

The pseudo code of the proposed genetic for service flow management procedure is shown in Fig. 5.

5. Implementation and experiments results

To test the proposed GA for service flow, we built our simulator based CloudSim simulator in Ref. [17]. The configuration of PC is as dual Core with 4 GB RAM with Windows 7 operating system. The proposed GA is developed on our simulator by JAVA. We also implemented a time optimization algorithm, Heterogeneous-Earliest-Finish Time (HEFT) [2], and Greedy Cost (GC) [7].

The HEFT algorithm is a list mapping algorithm which tries to allocate interdependent tasks at minimum execution time on a heterogeneous environment. But here in these paper it used to map interdependent services instead of tasks. The GC approach is to minimize workflow execution cost by assigning tasks to resources of lowest cost. Also it is implemented here for *FS* instead of workflow. The comparisons are performed based on the metrics of.

- The total cost
- schedule length ratio (SLR),
- speedup

Input: *FS and List of candidates*

Output: *the best mapping for FS on Candidates*

Steps:

1-Initialization part:

Threshold: determine the stopping criterion.

p: Population size indicates chromosomes number.

r: The percent of population replaced by Crossover.

m: mutation rate.

*p = Generate **p** chromosomes randomly*

***Evaluate:** For each chromosome ch in P, compute Fitness (ch).*

***Save** the max Fitness in Bsolution.*

2-Iterative part:

While (Bsolution < Threshold)

do

Create a new population p_n:

1. ***Select** (1 - **r).p** members from P and insert them into Ps using roulette wheel.*
2. ***Crossover:** select (r/2).p pairs of chromosomes from **p**. **For** each pair **(ch_1, ch_2) apply** order-based crossover to generate two offspring.*
3. ***Add** all offspring to p_n.*
4. ***Mutate:** select **m** percent of the chromosomes of p_n. Then apply insertion mutation method on them.*
5. ***Evaluate:** For each chromosome ch in P_n, compute Fitness (ch).*
6. ***Save** the max Fitness in Bsolution.*
7. ***Update:** p_n and p*

3- Finishing part:

Return the Bsolution.

Fig. 5. Proposed service flow management based genetic pseudo code.

The next experiments include different service flows (*FSs)* that include services from 10 to 100 services. The service flow is categorized as balanced structure or unbalanced structure as in Ref. [7]. In this paper balanced structure only is handled. Table 1 shows the selected best parameters of the proposed genetic algorithm (GA) that determined experimentally for service flow management.

The population size p of proposed GA is set to 20, the crossover rate is 0.8 and the mutation rate is 0.1, and the minimization of cost is selected as the target function for the proposed GA so it is running with a fixed number of iterations that is set to 100 as stopping criteria. We assume that the cost for each candidate is known and in this paper it assigned by a random value. The total cost of the proposed GA, HEFT and GC algorithms is presented in Fig. 6. The proposed GA consume less cost than HEFT and GC algorithms.

Table 1
Proposed GA selected Parameters.

The parameter	Selected value
p	20
r	.8
m	.1
iteration	100

Future Computing and Informatics Journal, Vol. 3 [2018], Iss. 2, Art. 18

Schedule Length Ratio (SLR): it is a key measurement of a scheduling algorithm based on makespan. The schedule length ratio (SLR) is defined by Eq. (2) as in Ref. [1].

$$SLR = \frac{Makespan}{\sum_{v_i \in CP_{\min}} min_{mt \in M}\{W_{i,t}\}} \tag{2}$$

The divisor is the summation of services minimum computation on the *FS*. Low value of SLR means that the better performance of mapping algorithm.

The SLR of the proposed GA, HEFT and GC algorithms is presented in Fig. 7. The proposed GA has less SLR than HEFT and GC algorithms that indicate the better performance of the proposed GA.

Speedup: The speedup value is defined as the ratio of the sequential execution time to the parallel execution time and can be computed by Eq. (3) as in Ref. [7].

$$Speedup = \frac{min_{mt \in M}\{\sum_{v_i \in V} W_{i,t}\}}{Makespan} \tag{3}$$

where numerator represents the sequential execution time computed by assigning all services to a single candidate and the parallel execution time here is represented by makespan.

The speedup of the proposed GA, HEFT and GC algorithms is presented in Fig. 8. The proposed GA has greater value of speedup measure than HEFT and GC algorithms that indicate the efficiency of the proposed GA.

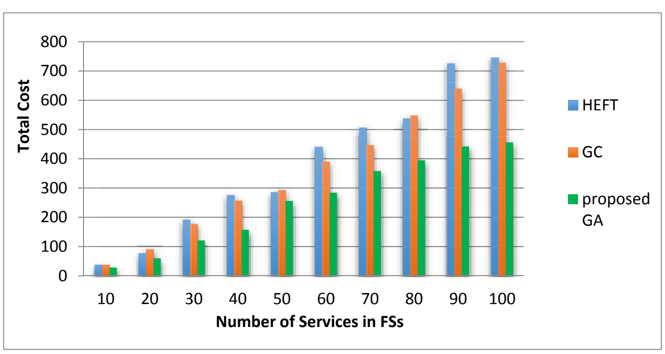

Fig. 6. Total cost of the proposed GA, HEFT and GC algorithms.

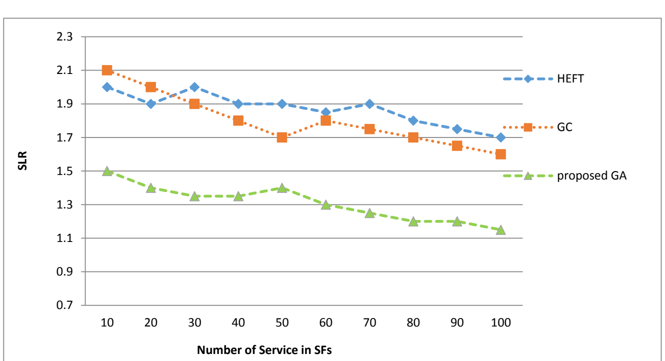

Fig. 7. Average SLR of the proposed GA, HEFT and GC algorithms.

Fig. 8. Average speedup of the proposed GA, HEFT and GC algorithms.

6. Conclusion and future work

This paper proposes the heterogeneous service flow management model that contains four modules likes service repository module, management module, broker module and SLA monitoring module. After that a genetic algorithm for service flow management is proposed. It facilitate service flow execution and management based service cost quality requirement in heterogeneous environments. Its main target is minimizing the total cost of service flow by taking into account the user specified budget. For testing the proposed genetic performance, our simulator based CloudSim simulator is developed. The experiments include different service flows that include services from 10 to 100 services. Some them are balanced service flow and the others are unbalanced. The selected parameters value of proposed genetic has been determined experimentally. The obtained results proves that the proposed genetic algorithm outperforms HEFT, and GC algorithms in terms of total cost, schedule length ratio and speedup measurements. The response time and security constraints and unbalanced structure may be handled in future work.

References

[1] Topcuoglu H, Hariri S, Wu M. Performance-effective and low-complexity task scheduling for heterogeneous computing. IEEE Trans Parallel Distr Syst 2002;13(3):260−74.
[2] Xiang Bin, Zhang Bibo, Zhang Lin. Greedy-ant: Ant colony system-inspired workflow scheduling for heterogeneous computing, vol. 5; 2017. p. 11404−12.
[3] Tracy DB, Howard JS, Noah B. Comparison of eleven static heuristics for mapping a class of independent tasks onto heterogeneous distributed computing systems. J Parallel Distr Comput 2001;61(6):810−37.
[4] Yu J, Buyya R, Tham CK. A cost-based scheduling of scientific workflow applications on utility grids. In: Proc. of the 1st IEEE International Conference on e-Science and Grid Computing, Melbourne, Australia; December 2005. p. 140−7.
[5] Wu AS, Yu H, Jin S, Lin KC, Schiavone G. An incremental genetic algorithm approach to multiprocessor scheduling. IEEE Trans Parallel Distr Syst September 2004;15(9):824−34.
[6] Zomaya AY, Teh YH. The, "observations on using genetic algorithms for dynamic load- balancing". IEEE Trans Parallel Distr Syst September 2001;12(9):899−911.
[7] Yu J, Buyya R. Scheduling scientific workflow applications with deadline and budget constraints using genetic algorithms. Sci Program J, IOS Press 2006;14:217−30.

[8] Yu J, Buyya R. Workflow scheduling algorithms for grid computing. Metaheuristics for scheduling in distributed computing environments. ISBN:978-3-540-69260-7. Berlin, Germany: Springer; 2008.
[9] Sakellariou R, Zhao H, Tsiakkouri E, Dikaiakos MD. Scheduling workflows with budget constraints. CoreGRID workshop on integrated research in grid computing. Technical report TR-05-22. Italy: University of Pisa; 2005. p. 347−57. November.
[10] Kim S, Weissman JB. A genetic algorithm based approach for scheduling decomposable data Grid applications. MD: ICPP.IEEE Computer Society: Silver Spring; 2004. p. 406−13.
[11] Ye G, Rao R, Li M. A multiobjective resource scheduling approach based on genetic algorithms in grid environment. In: Fifth International Conference on Grid and Cooperative Workshops, Hunan, China; 2006. p. 504−9.
[12] Pandey Suraj, Wu Linlin, Guru Siddeswara, Buyya Rajkumar. A particle swarm optimization (PSO)-based heuristic for scheduling workflow applications in cloud computing environments", technical report,CLOUDS-TR-2009-11,cloud computing and distributed systems laboratory. The University of Melbourne Australia; October, 2009.
[13] Talukde Khaled, Kirley Michael, Buyya Rajkumar. "Multiobjective differential evolution for scheduling workflow applications on global grids", concurrency and computation: practice and experience, vol. 21. New York, USA: Wiley Press; 2009. vol. 13, pp: 1742−1756.
[14] Xu Meng, Cui Lizhen, Wang Haiyang, Bi Yanbing. A Multiple QoS constrained scheduling strategy of multiple workflows for cloud computing. In: IEEE international symposium on parallel and distributed processing with applications; 2009. p. 629−34.
[15] Wu Zhangjun, Liu Xiao, Ni Zhiwei, Yuan Dong, Yang Yun. "A market-oriented hierarchical scheduling strategy in cloud workflow systems", journal of supercomputing, special issue on advances in Network&Parallel comptg, to be appeared. 2010.
[16] Mandal A, Kennedy K, Koelbel C, Marin G, Mellor-Crummey J, Liu B, et al. Scheduling strategies for mapping application workflows onto the grid. In: IEEE International Symposium on High Performance Distributed Computing (HPDC 2005); 2005.
[17] Ghalem B, Fatima Zohra T, Wieme Z. "Approaches to improve the resources management in the simulator CloudSim" in ICICA 2010, LNCS 6377. 2010. p. 189−96.