Cluster Comput (2014) 17:1013–1030
DOI 10.1007/s10586-013-0340-1

Trust-driven and QoS demand clustering analysis based cloud workflow scheduling strategies

Wenjuan Li · Jiyi Wu · Qifei Zhang · Keyong Hu · Jing Li

Published online: 23 January 2014

Abstract Due to the restrictions that most traditional scheduling strategies only cared about users' quality of service (QoS) time or cost requirements, lacked the effective analysis of users' real service demand and could not guarantee scheduling security, this paper added trust into workflow's QoS target and proposed a novel customizable cloud workflow scheduling model. In order to better analyze different user's service requirements and provide customizable services, the new model divided workflow scheduling into two stages: the macro multi-workflow scheduling as the unit of cloud user and the micro single workflow scheduling. It introduced trust mechanism into multi-workflow scheduling level. And in single workflow scheduling level, it classified workflows into time-sensitive, cost-sensitive and balance three types according to different workflow's QoS demand parameters using fuzzy clustering method. Based on it, it customized different service strategies for different type. The simulation experiments show that the new schema has some advantages in shortening workflow's final completion time, achieving relatively high execution success rate and user satisfaction compared to other kindred solutions.

W. Li (✉) · J. Wu · K. Hu · J. Li
Hangzhou Normal University, Hangzhou, China
e-mail: liellie@163.com

J. Wu
e-mail: dr_pmp@yahoo.com.cn

K. Hu
e-mail: hukeyong@yeah.net

J. Li
e-mail: Lijing_lijing@163.com

J. Wu · Q. Zhang
College of Computer Science and Technology,
Zhejiang University, Hangzhou, China
e-mail: zhangfee@zju.edu.cn

Keywords Cloud workflow scheduling strategy · Trust mechanism · QoS demand analysis · Fuzzy clustering · Customizable service

1 Introduction

Cloud computing uses internet as its foundation providing end users with the capability of using all kinds of resources on their demand at affordable price [1]. There is no doubt that cloud is the next generation's most potential application mode for sharing information resources [2–4]. Workflow is the kindred task combination [5]. In cloud, many large applications are usually consist of several kindred tasks which execute for the same goal and have data dependences and time order constraints. These tasks can be constructed into cloud workflow. Workflow mechanism offers an effective method for realizing resource sharing and cooperation between the multiple dynamic changing virtual organizations. Resource allocation and task scheduling is one of the key problems in realizing workflow since it directly influences workflow's execution efficiency. However, workflow scheduling is a NP-hard problem [6] which makes it very difficult to find a perfect solution.

Workflow scheduling quality of service (QoS) is one of the most important factors influencing cloud service quality. In cloud, users' QoS requirements generally can be divided into the performance QoS requirements and the trust QoS requirements. The performance QoS requirements often include the completion time (makespan), consumption fees and implementation accuracy, etc. And the trust QoS requirements are referred to the security of providing services and the possibility or reliability of access to services. However, current scheduling mechanisms are inclined to only attach importance to the performance QoS requirements while ignore the trust QoS requirements or to deal with the two aspects in isolation. In fact, malicious behaviors caused by service requester or provider lacking credibility will reduce the whole cloud service quality. For example, the abuses of service resources by malicious requesters will greatly consume service provider's resources and reduce its service capabilities; the frequent service failures and insecurity caused by dishonest providers will delay requester's tasks' execution and postpone the final completion time. Therefore, introducing trust mechanisms into task scheduling and paying attention to both performance QoS requirements and the trust QoS requirements are one of the key factors to improve the overall cloud service quality. For cloud computing systems whose target is to provide low-cost and on-demand services, the quality evaluation system obviously will be quite different from traditional environments. It seems more reasonable to provide cloud users with the services that closest to the quality they expect.

In response to the above issues, this paper introduced trust mechanism into cloud workflow scheduling process and proposed a novel trust-based customizable cloud workflow scheduling model on the basis of the preliminary work. The experimental data shows that the new model has certain advantages compared with other kindred models when used in the cloud environment. The main contributions of our work include: (1) introduce trust mechanism into workflow scheduling process; (2) divide cloud workflow scheduling into two different levels; (3) use fuzzy clustering method to classify workflows and better analyze cloud customer's service preference; (4) propose customizable single workflow scheduling algorithm under time and cost constraints.

The rest of this paper is organized as follows: part II briefly reviewed the related work, part III discussed cloud workflow scheduling model and part IV explained trust policy and how to use in scheduling. Part V introduced trust-based multi-workflow scheduling strategy and part proposed a novel customizable single workflow scheduling algorithm. Part was the design, results and analysis of the simulation experiments and the last part was conclusion and future work.

2 Related work

2.1 Trust related researches

Among the existing security solutions, trust has been proven to be a safe and high performance alternative security strategy in the distributed environments.

Traditional trust management models have two trust evaluation trends: (1) identity-based trust model which is through the exchange of trust certificates of the transaction parties to verify the authenticity and reliability of the identities; (2) behavior-based trust model which is through the observation of the cumulative historical transaction behavior and the feedback of other related entities to evaluate the credibility of the transaction partner.

In recent years, the typical trust models include: a dynamic trust prediction cognition model for the distributed systems was proposed in [7,8]. Pan et al. [9] put forward a reputation-based recommender discovery algorithm and through seeking for the credible recommender group and trust iterative calculation obtained the high reputation recommendation source. The feedback credibility based distributed trust model for the P2P environment was introduced in [10] claimed to be able to find really high recommendation credible nodes. Wang et al. [11] proposed a cloud model based subjective trust evaluation method. By using the trust changing cloud to record the changes of the objects' credit of trust, it more effectively supported the subsequent subjective trust decision-making process. Paper [12] introduced a deepness trust reasoning based service combination strategy to support the complex cross-organizational collaboration and its quality by the QoS evaluation and effective correction method. Wang et al. [13] established a new trust evaluation index system between cross-domains and also proposed a dynamic and comprehensive evaluation method for interoperability trust based on fuzzy variable weighting theory. A double incentive based trust and deception detection model called CCIDTM and an algorithm of conspiracy to deceive group testing to improve the model's dynamic adaptation was put forward in [14]. Since in trust and reputation systems in pervasive computingselfish users may maximize their profits by falsely declaring their recommendations strategically, thesis [15] introduced a Vickrey Clarke Groves mechanism based proof trust mechanism. A hierarchical trust model to meet heterogeneous subject trust requirement was proposed in thesis [16]. Long et al. [17] constructed trust reasoning and evolution system and a web service composition strategy to overcome the problem of poor quality of service in dynamic and complex internet. In addition, we have proposed several trust management model including the trusted domain based cloud trust management model and the fuzzy comprehensive evaluation based cloud trust management model for the cloud environment in our preliminary work [18–22].

The limitations of the traditional trust models lie in: (1) they are usually designed for a particular system, the establishment and implementation of trust is based on many assumptions and too restrictive conditions; (2) Trust has been studied alone and the lack of effective integration of trust with other systems mechanisms.

2.2 Workflow scheduling

Workflow can be described by directed acyclic graph (DAG) diagram or non-DAG method. Till now, most existing workflow scheduling researches are basing on DAG method.

Thesis [23] proposed an adaptive genetic based service workflow scheduling algorithm. The new algorithm constructed three kinds of models according to users' three main requirements (cost, completion time and security) in order to better adapt to the practical situation. A novel workflow resource allocation algorithm was put forward to improve the workflow execution success rate for grid environment through defining the key region and key region reliability in [24]. Yuan et al. [25] introduced a budget constrained grid workflows iterative heuristics with the objective of time optimization represented by DAG diagram. Since DAG-based workflow scheduling strategies cannot describe tasks' circular relations, there are also some scheduling researches based on non-DAG method. Cloud's tenet is to provide low-cost and on-demand services. So it emphasizes on the user's QoS demand. Thesis [26] present a scalable grid services QoS parameters structure model, defined grid service multi-dimensional QoS parameters utility update function and proposed an optimal grid workflow scheduling algorithm. Wang et al. put forward a service quality sensitive grid services workflow scheduling algorithm combining genetic and simulated annealing algorithm [27]. Maheswaran and coauthor [28] introduced QoS-driven resource management strategies into the large-scale multi-domain network computing systems and proposed a job-leveled dynamic scheduling algorithm to optimize QoS performance in the final execution period . Paper [29] improved the traditional Min-Min algorithm and put forward an adaptive QoS-driven Min-Min heuristic algorithm. A sufferage based Qsufferage algorithm was introduced to improve the makespan and average response time in [30]. Traditional QoS sensitive scheduling solutions tend to taking into account only time or cost requirements and neglect other demands. For example Buyya et al. [31] proposed three heuristic scheduling algorithms to solve the time cost optimization problems with cost or deadline (budget/deadline) constraints. But Buyya's solutions can only solve independent task set optimization.

Security is one of the key factors in influencing the scheduling success rate. So integrating trust to scheduling can improve its efficiency. Azzedin and Maheswaran [28] first introduced trust into grid resource management and proposed a novel job scheduling model and corresponding load minimization algorithm considering trust QoS demand. Humphrey did in-depth study on the trust conscious service computation model [32]. The existing research work on trust-based scheduling includes: Li et al. [33] put forward trust QoS driven grid task scheduling framework and related algorithms, thesis [34] introduced trust model and trust benefits function into grid service scheduling process and thesis [35] added trust computation into DLS algorithm and designed a trustable dynamic level scheduling model. Paper [36] put forward a trust-based grid workflow scheduling algorithm. A trust-based workflow scheduling method was proposed to ensure both QoS performance requirements and trust requirement in [37,38]. Han et al. [39] proposed a critical path and trust constrained based workflow task scheduling algorithm. Wang [40] in his doctoral thesis presented a reliability-driven trust model and basing on it proposed a look-ahead genetic algorithm LAGA to optimize both the time and reliability. In addition, we have put forward several scheduling solutions for cloud environment [41,42]

Although researchers have done a lot of work on task scheduling strategies and the service quality in the traditional Web environment, Cloud's unique features such as ultra-large-scale, resource completely virtualized and focusing on on-demand services make traditional solutions can not be used directly. The limitations of the traditional workflow scheduling models lie in: (1) they tend to ignore security or trust QoS demand in scheduling process which will decrease the efficiency of scheduling in the complicated cloud environment; (2) they are basically designed only for the micro matching of tasks to resources in the single workflow background and rarely consider the overall resource competition and allocation as the unit of workflow. While in the real network applications especially in the resources completely virtualized cloud environment, since service implementation details are shielded by providers, scheduling management program in the user side cannot directly bind micro tasks to the practical resource. So it seems more reasonable requesting and allocating resources as the unit of task set which make the computation of fees and other data accounting more easier; (3) they lack the reasonable analysis of different user's different QoS requirements and service preferences; (4) they tend to help task find the optimum service but not the most suitable service which may lead to the result that task bear extra expenses; (5) they usually only take into account one single optimization target, for example minimizing the finish time, minimizing the expenses or ensuring the reliability. While in reality user's QoS requirements are always fuzzy and complex and difficult to measure with one single indicator.

3 Definitions

In this part, we will firstly discuss trust conception and then define several useful models which will be used later.

3.1 Trust

Till now, there is no all researches' accepted trust definition. Based on in-depth study of trust and its previous researches, trust definition in this paper is as follows:

Definition 1 (Trust). Trust is trustor's recognition to trustee's identity and believe to his expected behaviors in a specific

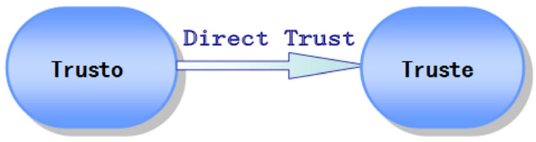

Fig. 1 Direct trust relationship

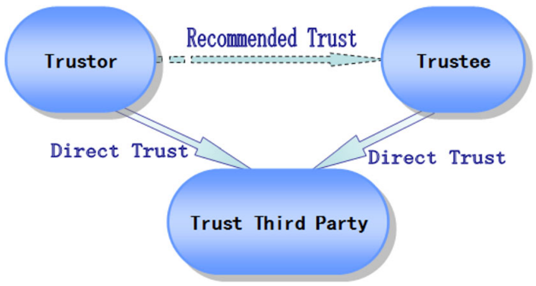

Fig. 2 Recommended trust relationship

time period and specific context. Trust is the subjective judgement of the trustor relying on his own experiences and other relevant knowledge. Trust's inspection indicators include authenticity, integrity, reliability and stability.

Definition 2 (Trust degree). It is the trust degree trustor to trustee which can also be called trust value or trust level. Trust degree can either be discrete (0,1) or be continuous (decimal between 0 and 1).

Definition 3 (Direct trust: DT). Direct trust comes from the direct transaction operation between trustor and trustee.

Definition 4 (Recommended trust: RT). Recommended trust is the indirect trust evaluation to trustee from the entities' that trustor's trust.

Definition 5 (Integrated trust: IT). Integrated trust is the comprehensive inspection on trustee's former transaction behaviors and the integrated evaluation combing both trustor's direct trust and other trust entities' recommendation trust.

Definition 6 (Cloud trust model: CTM). Cloud trust model represents the collection of trust relationships between cloud entities in cloud computing systems. It can be described as a five-dimensional vector CTM (P, C, A, R, F), where P represents the service provider set, C is the collection of the customers, A is the collection of trust properties, R means the trust relationship in the cloud systems and F represents the trust calculation of functions (Figs. 1, 2).

3.2 Cloud workflow model

Here, we use CWF $= \{g0, g1, g2, \ldots, gk - 1\}$ to represent workflow set and k $= |CWF|$ represents the number of workflows. The ith single workflow gi(i $\in [0, k-1]$) can be further described as gi $= \{gID, gUID, gTLimit, gCLimit, gEBW, gEStorage, gDAG\mathit{right}\}$, the meaning of each attribute is as follows:

- gID represents the workflow's identity number.
- gUID represents the cloud user who is the workflow's owner.
- gTLimit represents the execution deadline.
- gCLimit represents the expenses limitation.
- gEBW represents the workflow's expected bandwidth.
- gEStorage represents the workflow's expected storage capacity.
- gDAG represents the DAG diagram describing the relationships between tasks in the workflow.

3.3 Cloud provider model

Cloud providers are the resource owners who provide paid cloud service. Here P $= \{p0, p1, p2, \ldots, pk - 1\}$ represents provider set and k $= |P|$ represents the number of providers. The ith provider pi(i $\in [0, k - 1]$) can be further described as pi $= \{pID, pName, pHonest, pResourceSet\}$. The meaning of each attribute is as follows:

- pID represents the provider's identity number and pID is the unique identity of each provider in cloud system.
- pName represents the provider's name.
- pHonest represents the integrity of the provider. The value of service integrity is between [0, 1] and the greater the value the more likelihood the provider provide credible services.
- pResourceSet represents the resources set owned by the provider.

3.4 Resource model

pResourceSet $= \{r0, r1, r2, \ldots, rm - 1\}$ represents the resources belonging to one provider and m $= |R|$ means the number of resources. The ith resource rj (j $\in [0, m - 1]$) can be further described as rj $= \{rID, rCap\}$. The meaning of each attribute is as follows:

- rID represents the resource's identity number.
- rCap represents the capability of the resource. We use three-dimensional vector to depict the performance of the resources including computation power rComp, network transmission capability rBW and storage capacity rStor. So rCap $= \{rComp, rBW, rStor\}$.

3.5 Cloud services' QoS model

The QoS parameters used for evaluating quality of cloud services including response time, service bandwidth, expenses, reliability, security, accuracy, completeness and expansibility, etc. In order to better reflect cloud users' QoS requirements, we divides QoS model into QoS demand model and QoS provision model.

Definition 7 (Cloud QoS demand vector). It represents cloud users' requirements to the quality of cloud services using a five-dimensional vector to describe QoS requirements DQoS = {time, BW, storage, reliable, cost} on behalf of users' requirements to the finish time, bandwidth, storage capacity, reliability and the highest service expenses.

Definition 8 (Cloud QoS provision vector). It represents the quality of cloud services. Here it uses seven parameters to describe QoS provision quality PQoS = {AveComp, AveBW, MaxStorage, ServStability, CompPrice, BWPrice, StorPrice}. The parameters are the average computation power, average bandwidth, maximum available storage, service stability and the computation service, transmission, storage rented fees per unit.

3.6 Two-level based scheduling model

Figure 3 below shows how the two-level based scheduling model works.

In the user scheduling (multi-workflow scheduling) stage, first of all, cloud users (with their certain number of workflows) will send service request to the credible third party service intermediaries institution(in the figure, it is named USC). And the user scheduling center in the service institution will help to realize the binding of users to providers according current situation of user, resource and system load and then send the scheduling results to the corresponding users and providers. Users will then use their trust management module to evaluate the integrity of the providers in the recommendation list and choose the most suitable one to complete the transaction. Providers also will make trust judgment when asked for services. Single workflow scheduling takes place in the provider side. When one provider receives several users' service requests for executing their workflows, he will analyze the service preferences of the workflows and classify them into different types and then implement different strategies.

4 Trust mechanism in scheduling

4.1 The computation of trust

Since trust relationships are classified into direct trust and recommended trust, the computation of trust should also be treated differently according to its obtained method direct or indirect. For cloud entities, they should calculate integrated trust also. Current trust models always use simple weighted average method to compute the overall trust. The computation method can be abstracted as follows:

$$IT = \alpha \times DT + (1 - \alpha)RT \tag{1}$$

Here DT is the direct trust degree, RT is recommended trust degree, α and $(1-\alpha)$ is the weight of DT and RT respectively.

The key problem is how to set the weight of direct and indirect trust reasonably which has not yet been well resolved. What's more, to find an appropriate time attenuation function to measure the time influence to trust is also a key issue.

Below we will discuss the computation of trust.

(1) The computation of direct trust

Definition 9 (Trade slider window: TSW). It is the data-structure that records the direct recent transaction results. We use *TWLength* to represent the length of TSW.

The direct trust is obtained by direct transaction history between cloud entities. Formula 2 shows the computation method of direct trust (cloud entity A–B in transaction context tc).

$$DT(A, B, tc) = \frac{SuccessTradeNum}{TotalTradeNum} \tag{2}$$

Here *SuccessTradeNum* represents the success transaction times and *TotalTradeNum* means the total transaction times.

(2) The computation of recommended trust

When cloud entities want to trade with unfamiliar ones, they should use or combine recommended trust to aid the trust decision. Formula 3 below shows the method of compute recommended trust.

$$RT(A, B, tc) = \frac{\sum\limits_{p\in\Omega} DT(A, P, \mathrm{R}ecomm) * DT(P, B, tc)}{Length(\Omega)} \tag{3}$$

Here Ω represents the transaction recommendation set that entity trust.

(3) The computation of integrated trust

Definition 10 (Direct trust threshold: DTTh). It is the critical value using the direct trust only or using the integration of direct and recommended trust to compute trust degree. In this paper, when entities' direct transaction times are larger than *DTTh*, it can just use direct trust. *DTTh* can reduce the cost of trust decision.

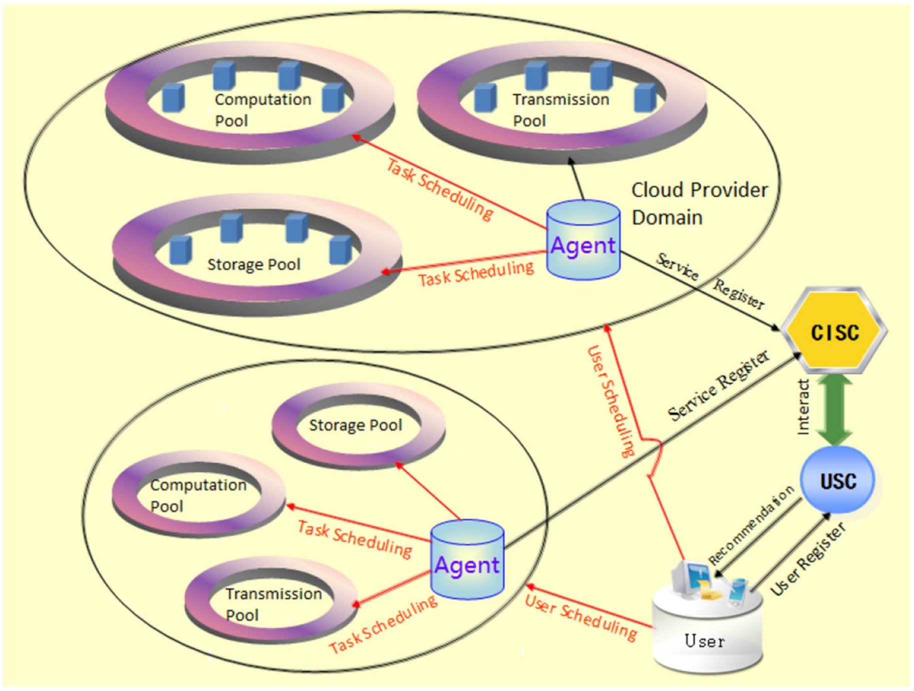

Fig. 3 Two-level based scheduling model

Definition 11 (Recommended credibility factor: RCF). It is the quantification of the trustworthiness of recommended trust. Below is its computation formula.

$$RCF(ttime) = \frac{DTTh - ttime}{DTTh} \tag{4}$$

Here *ttime* represents the actual transaction times in one *TSW* period. Obviously RCF changes with the actual transaction time.

Formula 5 shows how to compute integrated trust.

$$IT(A,B,tc) = \begin{cases} DT(A,B,tc) & ttime > DTTh \\ RT(A,B,tc) & ttime = 0 \\ \frac{1}{1+RCF} \times DT(A,B,tc) + \frac{RCF}{1+RCF} \times RT(A,B,tc) & 0 < ttime < DTTh \end{cases} \tag{5}$$

Since the recommended credibility factor (RCF) is forever between 0 and 1, it ensures that the direct trust's weight is always larger than the recommended trust's weight. In reality, people always count more on their direct judgment.

4.2 Trust decision

On the basis of dividing trust domains, this paper designed different trust decision strategies for cloud provider and customer. On the cloud provider side, trust domains are differentiated according to the current existing single cloud platforms and providers rely on their trust agents to manage trust relationship. While on the cloud customer side, they will judge independently or depend on the credible trust intermediary institutions to help them manage trust relationship.

(1) Trust implementation mechanism for customers

Figure 4 shows the main process of trust decision of cloud customers.

When A (user) want to trade with B (provider), A first check transaction history with B. If the effective number in the context of the trade context is larger than the direct trust threshold, then use the direct trust in the local trust table to take trust decision, else broadcast trust recommendation request within A's recommendation set and calculate integrate trust value combined with the direct trust.

(2) Trust implementation mechanism for providers

Since cloud providers mainly rely on their domain trust agent to help trust decision, the trust implementation mechanism is simpler.

Definition 12 (Reject trade list). It stores the entities that one reject to cooperate. Usually it means the entities cheat in the former transactions. Here we use ψ to represent the list.

When one provider is asked to provide services, he will ask trust agent in the domain for the judgments. And as long as the requester is not within the reject trade list, agent will

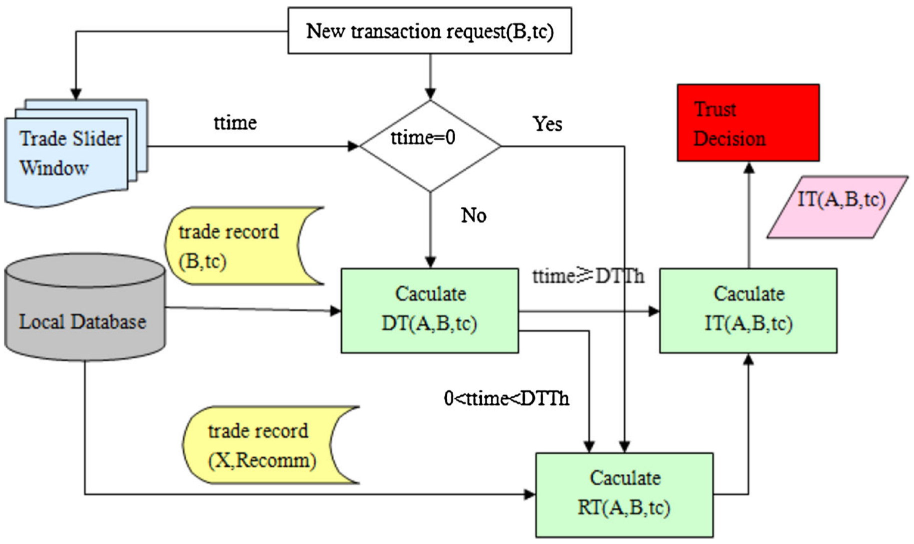

Fig. 4 Cloud customers' trust decision process

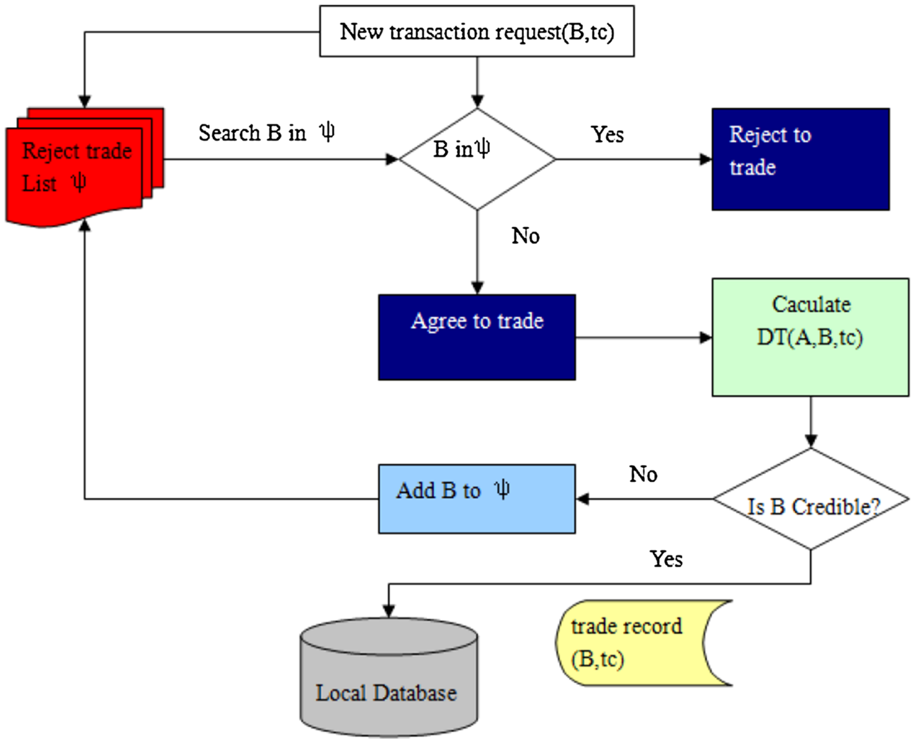

Fig. 5 Cloud providers' trust decision process

tell the provider to agree. After the deal, if the partner isn't credible, agent will add it into the reject trade list and avoid trading with it in the future. Figure 5 shows the trust process in provider side.

5 Trust based multi-workflow scheduling strategy

5.1 Trust filter in scheduling

When cloud user or provider receive the recommended list of transaction partners, they will use their trust evaluation module to make trust decision and choose the most suitable and credible ones to complete the transaction. Figure 6 below shows how trust mechanism works in the multi-workflow scheduling stage.

5.2 The similarity between service and demand

Multi-Workflow scheduling is the macro-scheduling level in our model. It realizes the virtual resource allocation as the unit of workflow set belonging to one cloud user.

Definition 13 (QoS tolerate price: QTP). It represents user's willing price for their demand service quality. QTP $=$ {BearCompPrice, BearBWPrice, BearStoragePrice}. The

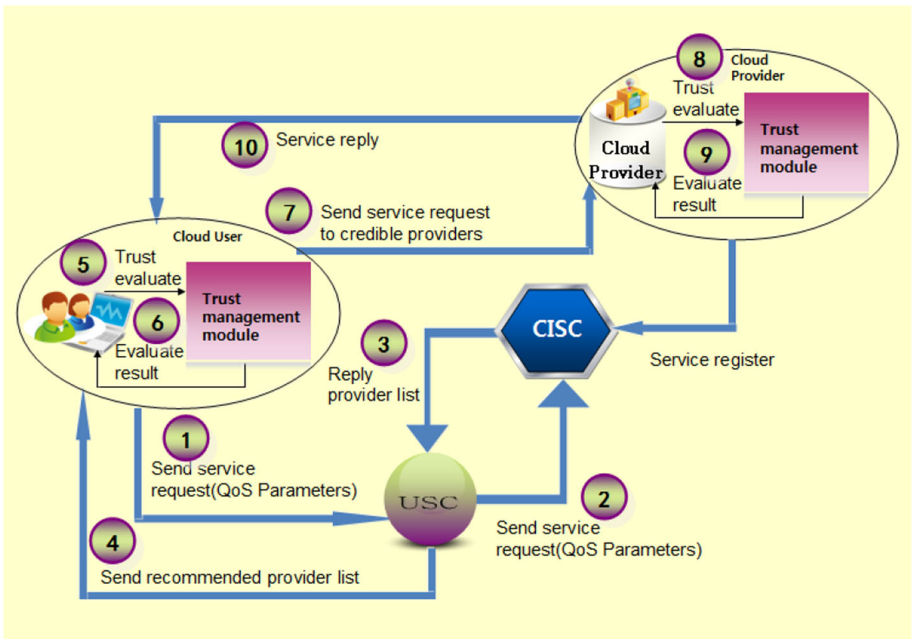

Fig. 6 Trust mechanism in multi-workflow scheduling stage

three parameters are bearing computation, bandwidth and storage fees. Obviously, *QTP* can be calculated through *DQoS*.

Definition 14 (QoS provision price: QPP). It represents provider's resource provision price for different service type. QPP = {CompPrice, BWPrice, StorPrice}.

Definition 15 (Service demand similarity: SDS). It represents the similarity between user QoS requirements to resources' provision quality. Here *SDS* is the Euclidean distance between *QTP* and *QPP*.

$$SDS = |QTP - QPP| \tag{6}$$

5.3 Trust-based multi-workflow scheduling algorithm

When more than one workflow (belonging to different cloud users) ask for cloud services at the same time, it binds the user (workflow) to the most similar provider's resource set in SDS to achieve the target of providing on-demand services.

Trust-based multi-targets QoS workflow scheduling algorithm named CMutiQoSSchedule is as follows:

Input: workflowList (workflows requesting for cloud services), providerList (cloud providers), tLevel (trust level in the transaction).

Output: scheduling results.

Main steps:

- Delete unqualified providers from providerList according to current trust level tLevel and generate reliable suppliers list named trustProviderList;
- Calculate each workflow's service tolerate price QTP in workflowList according to DQoS vector;
- Calculate each reliable provider's resource provision price QPP in trustProviderList according to PQoS vector;
- Calculate SDS using the Euclidean distance method and generate demand service similar matrix V.
- Search for each workflow its highest similarity provider and bind the workflow with the provider's resource set if the provider's trust is higher than trust threshold.

Trust-based multi-workflow scheduling algorithm uses trust mechanism to filter dishonest providers and then binds workflow to the provider whose service capability is closest to their demand. Assume that providers' number is m and workflows' number is n. Time complexity in the implementation of trust mechanism is o(m), calculating the service demand similarity matrix is o(m*n) and to bind the workflow with the appropriate provider is o(n). Therefore the overall time complexity of the algorithm is o(n*m).

6 Customizable single workflow scheduling strategies under time or expenses constraints

Single workflow scheduling is the micro scheduling stage realizing the binding of the real tasks belonging to one workflow to the actual resources obtained by the workflow.

6.1 The fuzzy classification of workflows

In order to better serve cloud users and meet different users' individual QoS requirements, we classify cloud user's workflow into expenses-sensitive, time-sensitive type and balance three types in single workflow scheduling level according to the workflow's parameters of service requirements using fuzzy clustering method. For expenses-sensitive users, it tries to reduce service cost under the premise of ensuring the latest completion time. While for time-sensitive type, it tries to shorten the completion time. And for middlebrow users, it uses compromise method.

Definition 16 (Task-resource suppose allocation matrix: SAM). It is the data structure that stores tasks' execution time and cost suppose it is distributed to the resources. The number of SAM's lines is the acount of the tasks and the number of columns is the acount of the resources. One element of SAM can be described as $SAM_{ij} = (t_{ij}, c_{ij})$ where t_{ij} and c_{ij} refer to task i's execution time and cost on resource j respectively.

Definition 17 (Task-resource allocation matrix: AM). It is the data structure that stores tasks' execution time and cost on their obtained resources. $AM_{ij} = (t_{ij}, c_{ij})$ means task i's execution time and cost on resources j.

Definition 18 (Cost/time adjust ratio: CTAR). It represents the ratio the time reduce value to the additional cost if a task want to shorten its completion time. Formula 7 shows the computation method of the ith task's *CTAR*.

$$CTAR_i = \frac{\Delta \cos t_i}{\Delta time_i} \times 100\,\% \tag{7}$$

The bigger the value of *CTAR*, the larger the extra expenses for a task shortening its completion time.

Definition 19 (Time/cost adjust ratio: TCAR). It represents the ratio the expenses reduce value to the extra time if a task want to reduce its cost. Formula 8 shows the computation method of the ith task's TCAR.

$$TCAR_i = \frac{\Delta time_i}{\Delta \cos t_i} \times 100\% \tag{8}$$

The bigger the value of *TCAR*, the longer the extra time for a task reducing its consumption cost.

Definition 20 (Cost time balance ratio: CTBR). It represents the preference of workflow's QoS requirments to time and cost. CTBR's calculation formula is as follows:

$$CTBR = \frac{|gCLimit - LowestCost| \times ShortestTime}{|gTLimit - ShortestTime| \times LowestCost} \times 100\,\% \tag{9}$$

Here, LowestCost means the possible lowest execution cost and gCLimit means the highest cost constraint. ShortestTime represents the possible shortest completion time and gTLimit represents the time deadline. When CTBR greater than one, it means the sensitivity of the workflow to time is greater than cost. When it is less than one, the sensitivity to cost is greater than time and when it is equal to one, time and cost are paid equal attention.

Definition 21 (Balance coefficient: BC). It represents the QoS similarity degree of time to cost. Formula 10 below shows the computation method.

$$BC = \begin{cases} 1000 & 0.999 < CTBR < 1.001 \\ \frac{1}{|CTBR-1|} & CTBR < 0.999, CTBR > 1.001 \end{cases} \tag{10}$$

The bigger *BC's* value, the more similarity workflow's QoS time demand to expenses. In this paper, it means the workflow is more appropriate to be a balance type.

Definition 22 (Request character vector: RCV). It represents the characters of one workflow to time, cost and balance used for fuzzy clustering. It can be described as following Formula 11.

$$RCV = \left\{ \frac{gTLimit}{glength}, \frac{gCLimit}{glength}, BC \right\} \tag{11}$$

The first two components represent the workflow's time and cost constraint in unit length. And several workflows' *RCVs* constitute request character matrix (RCM).

In this paper, we use fuzzy clustering method (FCM) to classify workflow into three types according to their different requirements. Fuzzy C-Means clustering method (FCM) was proposed by Bezdek [43] in 1981 and now FCM is one of the most important methods in fuzzy clustering area.

Workflow fuzzy classification algorithm named WFFuzzyClassify is as follows:

Input: workflowList(workflows requesting for cloud services), ε(cease threshold) and $MaxTimes$(maximum iterations).

Output: classification results.

Main steps:

(1) Initialize the sample matrix using RCM;
(2) Standardize the original data and compress the data to the interval of [0, 1];
(3) Classify the workflows into three categories randomly and initialize the fuzzy partition matrix U;
(4) Calculate the clustering center c_i, $i \in \{1, 2, 3\}$;
(5) Calculate objective function value each node to the clustering center. If it is small than cease threshold ε or the change from the last time is less than ε, the algorithm stops;
(6) Calculate the new division matrix U' and return to step (4).

Assume that resources' number is m, workflows' number is n and the maximum task number in workflow is x. Time

Table 1 Workflows' QoS requirements

gID	Time deadline (ms)	Cost constraints (cents)	Total length MI
g1	600	2,400	300
g2	450	1,500	150
g3	200	1,200	100
g4	1,000	5,000	800
g5	200	500	100
g6	400	1,000	250
g7	400	2,000	100
g8	800	3,000	500

Table 2 QoS demand per unit length

gID	Time	Cost
g1	2	8
g2	3	10
g3	2	12
g4	1.25	6.25
g5	2.5	5
g6	1.6	4
g7	4	20
g8	1.6	6

complexity in generating task-resource suppose allocation matrix SAM and allocation matrix Am are both o(x*m). Assume the critical path of workflow is known, the time complexity of the initialization process of workflow fuzzy classification algorithm is o(n*x*m) and the fuzzy classification process is o(n*Maxtimes). Here Maxtimes represents the maximum iteration times and can be customized.

Following we use an example to show how the fuzzy classification mechanism works. Assume that there are eight workflows now in the cloud system. Table 1 shows workflows' QoS requirements.

Table 2 shows workflows' time and cost constraints per unit length.

Assume that in the current system resource situations, workflows' shortest completion time and lowest cost are shown in Table 3.

Using the datas in Tables 1 and 3, each workflow's CTBR and BC can be calculated out. Table 4 shows the results.

From Tables 2 and 4, RCM can be found. Figure 7 below shows the result.

The original data and data after standardized treatment is shown in Fig. 8.

The function of data standardization is:

$$g'ij = (gij - \overline{g}j)/Sj \tag{12}$$

Here, $\overline{g}_j$ is the mean value of time request by workflows and S_j is standard deviation of time demand. Then we use

Table 3 Shortest time and lowest cost

gID	Shortest time (ms)	Lowest cost (cents)
g1	400	1,200
g2	300	1,000
g3	180	800
g4	800	2,500
g5	150	500
g6	320	900
g7	350	1,000
g8	600	2,000

Table 4 Workflows' CTBR and BC

gID	CTBR	BC
g1	2	1
g2	0.4	1.67
g3	4.5	0.29
g4	4	0.33
g5	0	1
g6	0.44	1.79
g7	7	0.17
g8	1.5	2

$$\begin{pmatrix} 2 & 8 & 1 \\ 3 & 10 & 1.67 \\ 2 & 12 & 0.29 \\ 1.25 & 6.25 & 0.33 \\ 2.5 & 5 & 1 \\ 1.6 & 4 & 1.79 \\ 4 & 20 & 0.17 \\ 1.6 & 6 & 2 \end{pmatrix}$$

Fig. 7 Request character matrix

$$\begin{pmatrix} 2 & 8 & 1 \\ 3 & 10 & 1.67 \\ 2 & 12 & 0.29 \\ 1.25 & 6.25 & 0.33 \\ 2.5 & 5 & 1 \\ 1.6 & 4 & 1.79 \\ 4 & 20 & 0.17 \\ 1.6 & 6 & 2 \end{pmatrix} \longrightarrow \begin{pmatrix} -0.286 & -0.187 & -0.044 \\ 0.905 & 0.224 & 0.941 \\ -0.286 & 0.634 & -1.088 \\ -1.179 & -0.546 & -1.029 \\ 0.310 & -0.803 & -0.044 \\ -0.762 & -1.008 & 1.118 \\ 2.095 & 2.277 & -1.265 \\ -0.762 & -0.598 & 1.426 \end{pmatrix} \longrightarrow \begin{pmatrix} 0.273 & 0.250 & 0.454 \\ 0.637 & 0.375 & 0.820 \\ 0.273 & 0.500 & 0.066 \\ 0 & 0.141 & 0.088 \\ 0.455 & 0.062 & 0.454 \\ 0.127 & 0 & 0.886 \\ 1 & 1 & 0 \\ 0.127 & 0.125 & 1 \end{pmatrix}$$

Fig. 8 Data standardized process

range transformation method to transform data in the matrix into the closed interval [0, 1] values. Fomula 13 shows the calculation method.

$$g''ij = (g'ij - g'j\min)/(g'ij\max - g'ij\min) \tag{13}$$

After setting threshold and iteration calculations, the eight workflows are divided into three clusters. That is Cluster0 = {g7}, Cluster1 = {g4, g5, g6, g8}, Cluster2 = {g1, g2, g3}.

Cluster0 is marked the time-sensitive workflow set, cluster1 the cost-sensitive set and cluster2 balance type set.

6.2 Customizable single workflow scheduling strategy

This paper distinguishes different workflow types and adopts different scheduling strategies.

(1) Time-sensitive workflow scheduling strategy under cost constraints

For time-sensitive workflow, it is advisable to try best to shorten the completion time on the premise of ensuring the highest cost constraint. The corresponding scheduling strategy is as follows:

(a) Initialize the shortest completion time stime = MAXINT and the execution expenses cost = 0;
(b) Generate the SAM according to the tasks' parameters in the workflow and the obtained resources' performance;
(c) Find all critical paths and put all critical paths in KeyPathList according to the workflow's DAG diagram;
(d) If KeyPathList is not NULL, pick one critical path up, bind the tasks in the path to the resources gaining the earliest completion time and assign all the non-critical tasks to the resources having the lowest cost. Generate AM. If all critical paths are traveled, go to step (i);
(e) Calculate the total cost and completion time according to AM. If $\cos t > MaxCost$, go to step (f). Else determine whether $stime > time$. If so, let $stime = time$, go to steps (d);
(f) Compute CTAR when the critical tasks running on other resources according to SAM and sort the tasks in CTAR's descending order;
(g) Reassign tasks to the longer execution time resource in turn to reduce the cost;
(h) Calculate the new completion time and the change of cost $\Delta \cos t$ after resource re-allocation. If $\Delta \cos t < |\cos t - MaxCost|$, return to step (g), else determine whether $stime > time$, if so, let $stime = time$, return to step (d);
(i) Return stime and the task-resource distribution relationship.

(2) Cost-sensitive workflow scheduling strategy under time deadline constraints

For cost-sensitive workflow, it is advisable to try best to compress execution expenses on the premise of ensuring the time constraints. The corresponding scheduling strategy is as follows:

(a) Initialize the minimum cost scost = MAXINT and the completion time time = 0;
(b) Generate the SAM according to the tasks' parameters in the workflow and the obtained resources' performance;
(c) Assign the tasks to the lowest cost resources according to SAM and generate distribution matrix AM;
(d) Calculate the total cost and completion time according to AM. If $time \le MaxTime$ go to step (h), else go to step (e);
(e) Calculate TCAR when tasks are assigned to expensive while faster resources and sort tasks in TCAR's descending order;
(f) Reassign tasks to the more expensive but faster resources in turn to shorten the completion time;
(g) Calculate the new execution cost and the total change of time $\Delta time$ after resource re-allocation. If $\Delta time < |time - MaxTime|$, return to step (f), else go to step (h);
(h) Let scost = cost, return scost and the task-resource distribution relationship.

7 Experiments

7.1 Scene simulation experiments based on Netlogo [44]

We designed cloud workflow scheduling scene simulation experiments to compare the difference between loading and unloading trust mechanisms. NetLogo is developed by Uri Wilen.sky of Northwestern University in1999 and perfected and further developed by Center for Connected Learning in the university. NetLogo is computer-based modeling and simulation software which is able to simulate the natural or social systems especially suitable for the simulation modeling of the complex system evolution with time. The scene of NetLogo includes turtle, patch and observer. Patches are the static background simulating various ecological scenes of the real world such as the grasses, road etc. Turtles are the main role of the simulation scene. Turtles can move on the patches, can breed and die. They are the real dynamic changing entities on the scene. Different role turtles can be added into the same scene to represent different simulation entities. Observer is an independent third party who observes the scene.

In our simulation experiments, we use red-colored circle turtle to represent cloud customer while yellow square turtle to represent providers. Blue edges are used to represent the ordinary trade relationship between customers and providers, yellow edges represent the recommended trade relationship and red edges the cooperation relationship between providers.

(1) Experiments results when not loading trust mechanism

Simulation steps are as follows:

(a) Generate designated number of cloud customers and providers and randomly deploy their performances. For

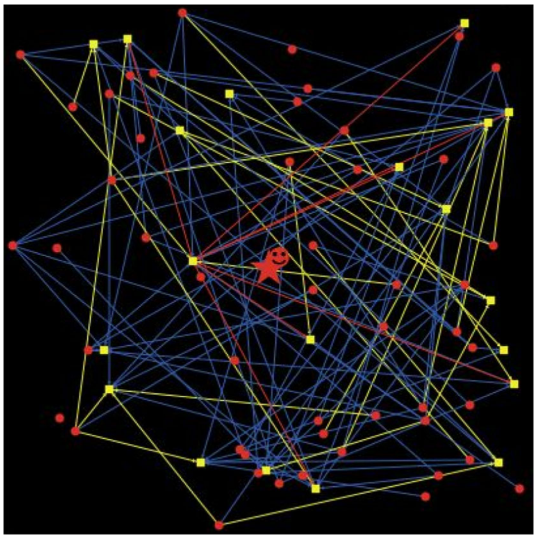

Fig. 9 Transaction relationship in the initial time

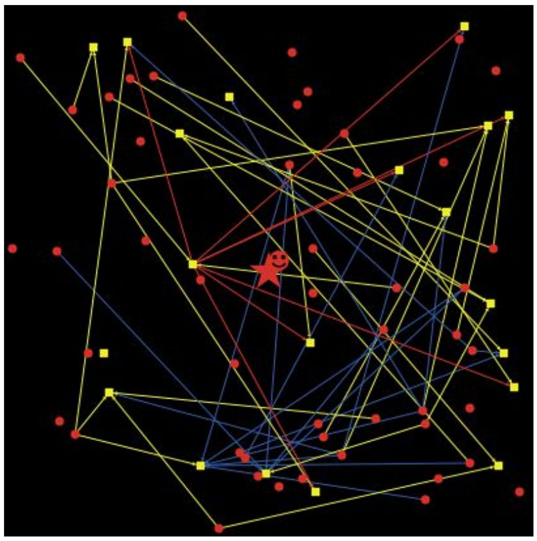

Fig. 10 Transaction relationship after trading

customers it represents their integrity and for providers it represents their service reliability.

(b) Generate the system's entity relationship network randomly including the history transaction links between customer and provider or between different providers.

(c) Each customer turtle completes 100 times cloud transactions. Everytime it will first choose one familar provider in this trade context to complete the trade. If there are no familar ones in this trade context, it will trade with the provider who is recommended by other providers. If the recommendation one cannot be found also, it will choose a random provider to finish the transaction. If the deal is success, an transaction edge will be increased between the client and the provider if there does not exist one before. If the transaction fail, the edge disappears if it exists and the recommendation edge will also disappears if the provider is got by recommendation.

(d) Each provider turtle completes 100 times trading. It will firstly choose one familar other providers to cooperate. If it cannot find one, it will randomly select one to finish the transaction. After the deal, if it succeeds, a cooperation edge between the two providers will be increased if there does not exist before the trade. If the deal fails, the edge will disappear if there originally exists.

Figures 9 and 10 show the simulation experiment results. Figure 9 shows the randomly generated cloud trading relationships in the initial time. Figure 10 shows the latter relationship network after 100 times transaction each customer turtle and provider turtle. For better observation, it chose to analyze cloud system that contains 50 cloud customers and 20 providers. From the results we could see that transaction relationship network gradually shrunk when there existed malicious nodes and no trust mechanism was introduced. In the real cases, it gives expression to the phenomenon that entities gradually lose the faith of using cloud systems after several transaction failures.

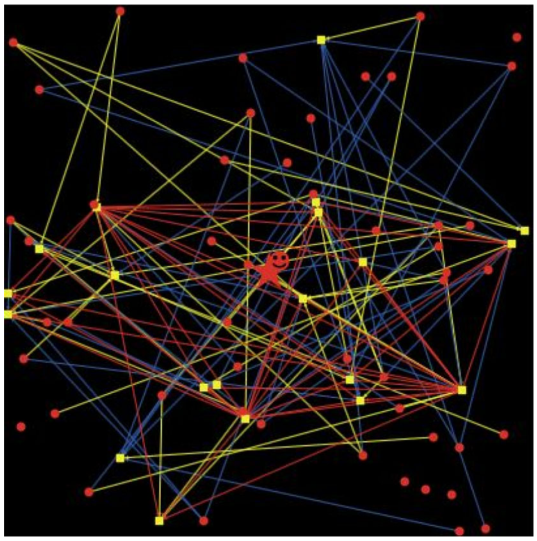

Fig. 11 Transaction relationship in the initial time

(2) Experiments results when loading trust mechanism

Figures 11 and 12 show the scene simulation experiment results when we loaded the proposed trust mechanism into cloud transaction process. Figure 11 shows the randomly generated cloud trading relationships in the initial time. Figure 12 shows the latter relationship network after 100 times transaction each customer turtle and provider turtle.

The results show that through the loading of trust mechanism and taking trust evaluations to trading partners before

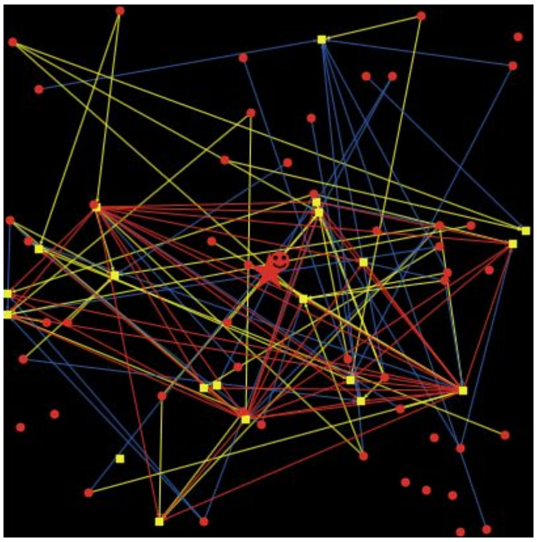

Fig. 12 Transaction relationship after trading

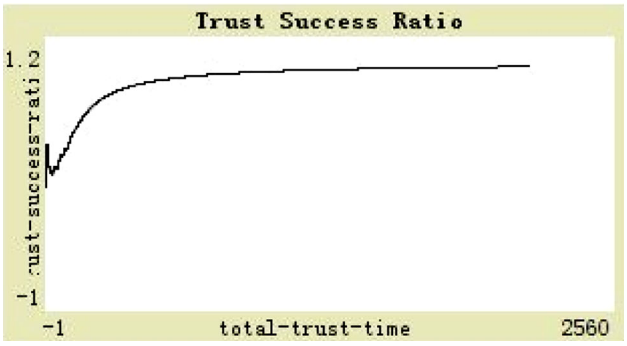

Fig. 13 The result of trust accuracy

transactions, it better protects cloud entities' security, better maintain the stability of entities' relationship and also it improves the transaction success rate.

(3) Performance evaluation of our model

Using NetLogo's reporter, we evaluated the performance of the proposed model. Performance analysis includes two items: trust accuracy and transaction success rate. Trust accuracy is the ratio the probability of obtaining collect trust decisions. Transaction success rate is the success trading proportion. Figures 13 and 14 show the results of the performance analysis. Figure 13 shows the result of trust accuracy and Fig. 14 is the result of transaction success rate.

The experimental results show that the model's trust accuracy and trade success rate maintain a high level when proposed trust mechanism loaded except in the early chaos.

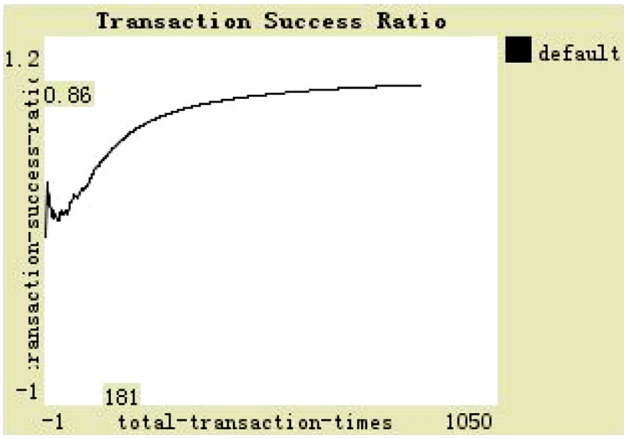

Fig. 14 The result of transaction success ratio

7.2 Scheduling performance test based on CloudSim

(1) The design of simulation platform

This paper also set up a simulation platform basing on CloudSim [45,46] to test the performance of the new cloud workflow scheduling model. CloudSim is the most famous cloud simulation tool that designed by the Grid laboratory in Melbourne University and Gridbus project. Researchers can do a lot of cloud simulation work including task scheduling on CloudSim. But at present, CloudSim cannot realize workflow scheduling and trust strategies simulation directly. So this paper extended CloudSim, made it support workflow simulation and increased trust validation in the simulation process. We added several simulation entities into CloudSim according to the new model's structure and rewrote relevant message simulation process according to the new realization mechanism. Our experiment's environment is P2.8 GHz CPU, 4GB memory and Windows 7 operating system.

The simulation system contained cloud interactive environment, user generator, workflow generator, provider generator, resource generator and trust evaluator several parts. Cloud interactive environment is mainly used to realize the interaction of different cloud entities. User generator generates user nodes randomly and deploys all kinds of parameters including trust, demand service requirements and so on. Workflow generator randomly generates cloud tasks and the workflow's DAG diagram. Provider generator generates cloud providers according to operator specified number and configures their parameters including service integrity, resource nodes and performance. Task scheduler is applied to deploy and implement the workflow scheduling strategy. Trust evaluator is used to increase trust mechanism in the multi-workflow scheduling level to help cloud transaction decision.

Table 5 Parameters of providers

PID	Computation capability (MIPS)	Transmission capability (MB)	Storage capacity (MB)
1	2×10^6	10^4	2×10^6
2	6×10^6	2×10^4	10^6
3	10^6	2×10^6	10^6
4	4×10^6	10^4	2×10^6
5	2×10^6	10^4	10^6
6	10^6	2×10^4	10^6
7	4×10^6	2×10^4	4×10^6
8	10^6	10^4	2×10^6
9	4×10^6	10^4	10^6
10	6×10^6	10^4	10^6

Table 6 Parameters of resources

Resource type	Computation capability (MIPS)	Transmission capability (MB)	Storage capacity (MB)
Computation	10,000–50,000	30–150	400–2000
Storage	1,000–5,000	30–150	4,000–20,000
Bandwidth	1,000–5,000	300–1,500	400–2000

Table 7 Parameters of cloudlets

Cloudlet type	Length (MI)	Expect bandwidth (MB)	Expect storage (MB)
Computation	5,000–20,000	10–50	25–125
Storage	500–2,000	10–50	250–1,250
Bandwidth	500–2,000	100–500	25–125

(2) Simulation environment

Simulation experiments simulated cross-clouds platform that contained ten cloud providers, 150 virtual resource nodes (randomly belonged to a certain provider) and a certain number of customers. Table 5 shows the parameters of the providers. The parameters of the virtual resources are shown in Table 6. The parameters of different cloudlet types are shown in Table 7.

(3) Introduction to related algorithms

Simulation experiments mainly compared our model with classical heuristic table scheduling algorithm DLS [47] and MCP [48], Trust-QoS Enhanced Grid Service Scheduling algorithm [34], TD Min–Min [49] and Berger's Model [50].

Scheduling can be classified into static scheduling and dynamic scheduling. Most current research work is static scheduling based and majority static scheduling models use table-based technique. The kernel of table-based algorithm is: sort the tasks using priority and schedule the task in order. MCP and DLS are both heuristic table scheduling solutions. MCP algorithm uses task's (node's) latest start time as late as possible as the priority, arranges task in ascending order and schedules task to the earliest start machine using insertion method. DLS algorithm uses the distance between the latest start time to the earliest start time as the priority named dynamic level (DL) and schedules tasks according to DL's decending order.

Based on the grid trust model and trust utilization functions, Zhang studied trust-driven grid job scheduling mechanisms and proposed trust-driven (TD) Min–Min and a novel trust-QoS enhanced heuristic based on trust relationship. Berger's Model [27] is one of the few scheduling models designed specially for cloud environment. The Berger's Theory comes from the sociology field which is about the fair distribution of the society. Berger's Model treats the justice distribution of user tasks as its' primary goal. Through the binding of specific user's tasks to specific resources, it attempts to achieve the goal of meeting the greatest consistency of user expectations and the actual allocation of resources.

(4) Simulation experiments to test the influence of trust strategie in workflow scheduling

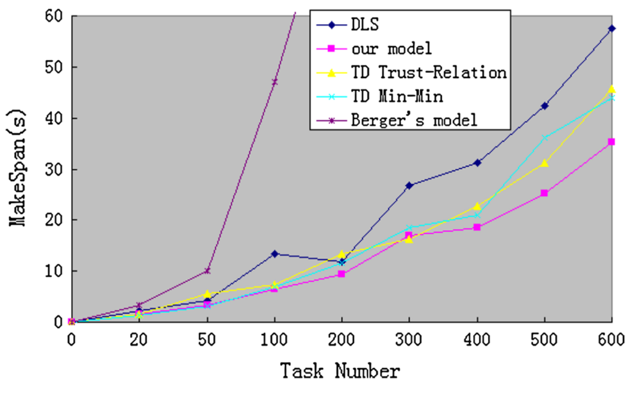

Fig. 15 Comparison of final completion time (v = 0.25)

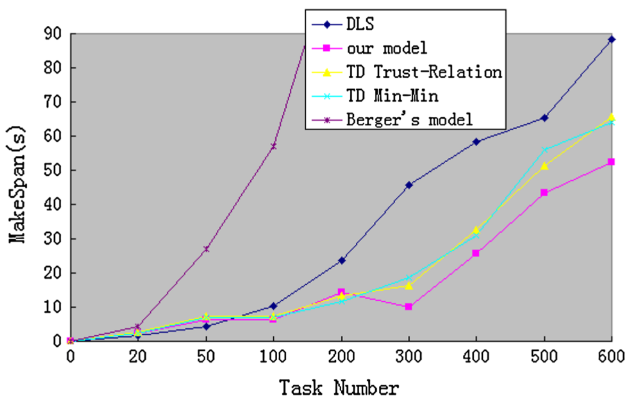

Fig. 16 Comparison of final completion time (v = 0.5)

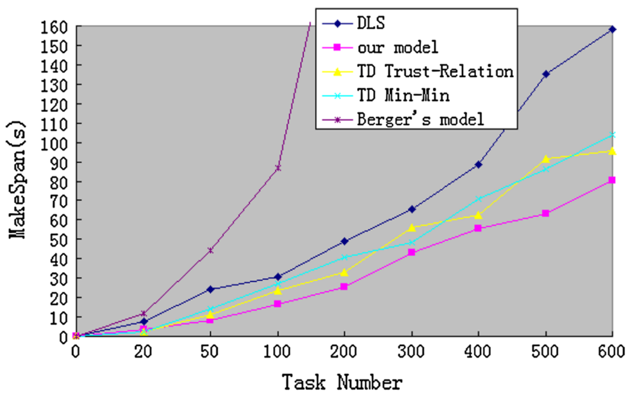

Fig. 17 Comparison of final completion time (v = 0.75)

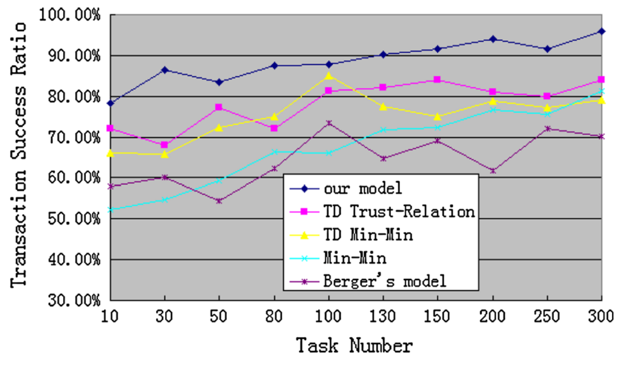

Fig. 18 Comparison of transaction success ratio (v = 025)

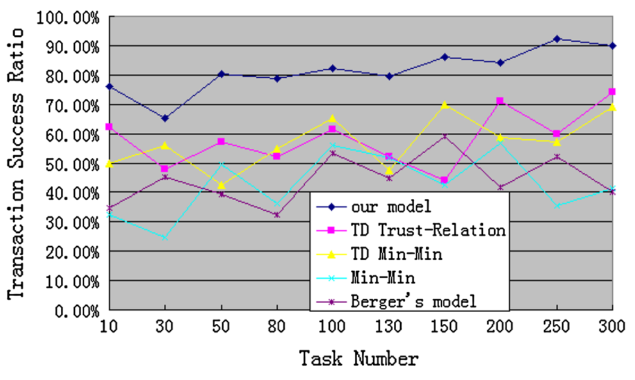

Fig. 19 Comparison of transaction success ratio (v = 05)

In cloud, most providers will perform their service duties in accordance with their promise. But also there are a few dishonest providers who may refuse to provide services, may provide mendacious services or even cheat in services. In order to better reflect all kinds of possible behaviors in the cloud, the simulation experiments added in a percentage of malicious provider nodes.

The comparison experiments' result about workflow's final completion time is shown in Figs. 15–17:

The result indicates that the new trust based scheduling model shorten the workflow's final completion time compared to the traditional model MCP and DLS which do not load trust mechanisms. MCP and DLS try to shorten the execution time of critical tasks on the critical path by assigning them to the shortest execution time resources. But they ignore resources' integrity which possibly leads to the result that key tasks are sent to dishonest resources, delayed their execution and needed to be re-scheduling. So the workflow's final completion time is postponed. While our model took into account the trust included comprehensive performance of the candidate resources in the critical tasks' scheduling process and the probability of failure scheduling is low which succeeded in shortening the whole workflow's execution time. With task number rising, trust-based workflow scheduling model shows more advantage.

The result of the comparison experiments on tasks transaction success ratio is shown in Figs. 18–20.

The result shows that: in cloud systems with some malicious nodes, trust mechanism embedded workflow scheduling model can ensure higher transaction success rate since it efficiently eliminates dishonest providers according to former transaction data and other credible nodes' trust recommendation.

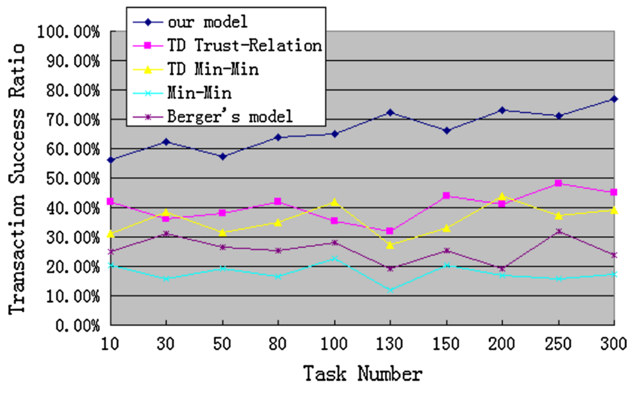

Fig. 20 Comparison of transaction success ratio (v = 0.75)

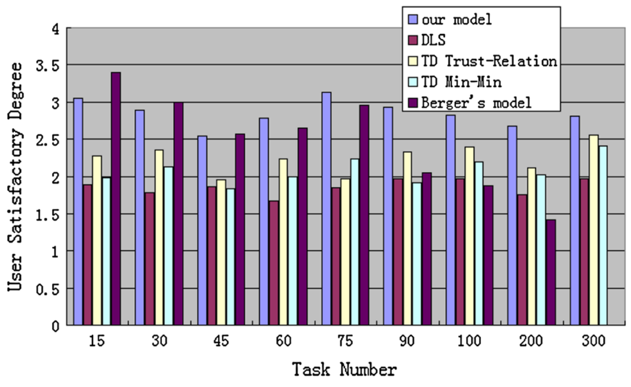

Fig. 21 Comparison of user satisfactory (v = 0.25)

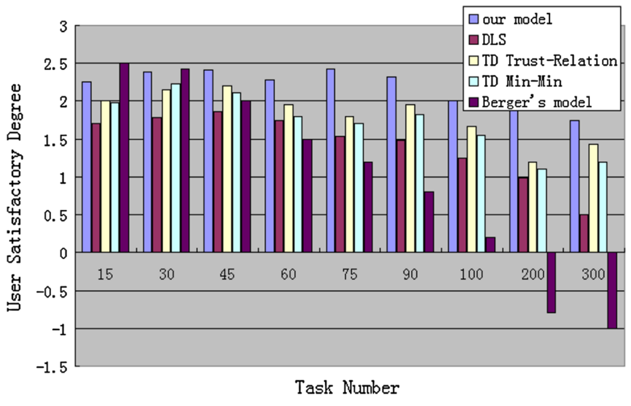

Fig. 22 Comparison of user satisfactory (v = 0.5)

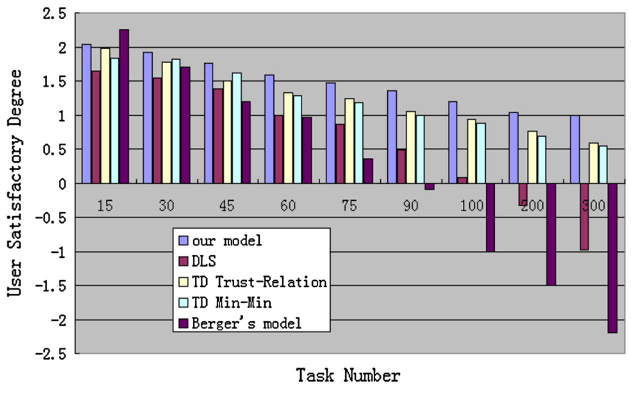

Fig. 23 Comparison of user satisfactory (v = 0.75)

(5) Simulation experiments to test users' QoS satisfaction degree

Cloud's aim is to provide cheap and on-demand service, so it pays great emphasis on the user's QoS requirements. And the goal of workflow scheduling is to improve cloud user satisfaction through the reasonable distribution of resources to tasks. User satisfaction refers to the identical degree users' actual obtained resource level to their expecting resource level. Since we only consider three aspects of QoS requirements: time, expenses and the reliability, basing on users' QoS demand model, single user's satisfaction degree can be calculated as follows:

$$Jvalue_i = \ln\left(\frac{\rho(Etime_i)}{\underset{k\in ORSet_i}{Min}\ (time_k)} + \frac{\omega(E\cos t_i)}{\sum_{k\in t ORSet_i}\cos t_k} + \frac{\tau(Ereliable_i)}{\sum_{k\in t ORSet_i} trust_k/|ORSet_i|}\right) \tag{14}$$

$Etime_i$, $Ecost_i$ and $Ereliable_i$ refer to the ith user's expectations to time, cost and resources' reliability respectively. *ORSet* means the user obtained resources set and ρ, ω, τ refer to the experience coefficient to demand time, money and reliability respectively. The total user satisfactory degree is shown in Formula 15.

$$UJvalue = \frac{\sum_{i=UserSet} Jvalue_i}{n} \tag{15}$$

Here, *UserSet* means cloud user set and n is the number of users. User satisfactory degree simulation experiments mainly compared our model with DLS and Berger's model. The results are shown in Figs. 21–23.

The time consumption of Berger's model is large. When task number rises, time cost rises rapidly. So the simulation experiments are divided into the two segments: small scale (task number is between 0 to 120) and larger scale (task number is larger than 120). In the small scale it compared three algorithms and in the large scale it compares two algorithms. From the results we can see: when the scheduling problem scale is small, the effect of our model and Berger's is identical in the performance of user satisfaction degree better than DLS; when in a larger problem scale, our model performs better. Since DLS scheduling algorithm only pays attention to the time QoS requirement of tasks especially critical tasks while ignores other QoS requirements, it cannot guarantee the overall user satisfaction. Berger's model assigns tasks to the shortest Euclidean distance resources according to tasks' demand so as to achieve the maximum consistency user's resource expectation to their actual allocated level. So the overall satisfaction of customers of Berger's model is high. But it can't guarantee the whole system's throughput and scheduling cost is huge which lead to the difficulties in the actual system realization. The new model divides cloud workflow scheduling into two levels and adopts different strategies according to cloud user's QoS demand preference which reduces the scale of the scheduling problem and reflects the real needs of cloud users.

8 Conclusion and future work

In order to better reflect cloud's tenet providing low-cost and on-demand services and embody cloud user's QoS requirements, this paper proposed a novel trust based cloud workflow scheduling model. Trust mechanism ensures the execution success rate of critical tasks and the introduction of two-level based scheduling mode reduces the problem scale of workflow scheduling. Multi-workflow scheduling helps to realize the QoS requirements analysis and service customization as the unit of user. Fuzzy clustering based user service preference classification and workflow scheduling strategy improves the workflow's overall operation efficiency and user satisfaction. In order to verify the performance of the new trust model, we did several simulation experiments on NetLogo and the results show that loading trust mechanism can effectively improve the transaction success rate in the cross-cloud environment, while also be better able to maintain cloud entities' trust relationship network. In addition, we designed a workflow scheduling simulation platform based on CloudSim. By adjusting the proportion of malicious nodes, we compared our model with other classical scheduling algorithms DLS, TD Min–Min, TD Trust-Relation and Berger's model in makespan, execution success rate and user satisfaction. The final results also show that our model has certain advantages in all the above indicators.

However, the testing and application of our program has still been in the simulation experimental stage till now. So in the future, we plan to build a small cloud prototype system and realize the deployment of our strategy in the real environment to test its efficiency and effectiveness. Furthermore, research on cloud services composition and composition scheduling policy, cloud service billing and auditing strategy is also our future work.

Acknowledgments This work is supported by the Natural Science Foundation Research Project of Zhejiang Province (LQ12G02016). The authors also gratefully acknowledge the helpful comments and suggestions of the reviewers.

References

1. Liu, P.: Cloud Computing, 2nd edn. Publishing House of Electronics Industry, Beijing (2011)
2. Foster, I., Kesselman, C.: "The Grid: Blueprint for a New Computing Infrastructure". Morgan Kaufmann, San Francisco (1999)
3. Chinese Cloud Computing Forum. http://bbs.chinacloud.cn
4. Vouk, M.A.: Cloud computing-issues research and implementations, Proceedings ITI 2008 30th International Conference on Information Technology Interfaces, 2008. IEEE, (2008)
5. Zhu, F., He, Y.: Parallel Scheduling Algorithm in Distributed Computing Theory and Design. Wuhang University Press, Wuhang (2003)
6. Ullman, J.D.: Complexity of sequencing problems. In: Coffman, E.G. (ed.) Computer and Job Scheuling Theory. Wiley, New York (1976)
7. Li, X., Gui, X.: Cognitive model of dynamic trust forecasting. J. Softw. **21**(1), 163–176 (2010)
8. Li, X., Gui, X.: Research on dynamic trust model for large scale distributed environment. J. Softw. **18**(6), 1510–1521 (2007)
9. Pan, J., Xu, F., Lv, J.: Reputation-based recommender discovery approach for service selection. J. Softw. **21**(2), 388–400 (2010)
10. Hu, J., Wu, Q., Zhou, B., et al.: Robust feedback credibility-based distributed P2P trust model. J. Softw. **20**(10), 2885–2898 (2009)
11. Wang, S., Zhang, L., Li, H.: Evaluation approach of subjective trust based on cloud model. J. Softw. **21**(6), 1341–1352 (2010)
12. Hu, C., Chang, B., Xu, H., Zhao, Q.: Approach of service combination based on deepness trust reasoning in complex cross-organizational collaboration. J. Central South University (Science and Technology) **43**(2), 567–575 (2012)
13. Wang, Y., Xiao, X., Jia, L.: Dynamic and comprehensive evaluation method for interoperability trust based on fuzzy variable weighting. J. Comput. Res. Dev. **49**(6), 1235–1242 (2012)
14. Xie, X., Liu, L., Zhao, P.: Trust model based on double incentive and deception detection for cloud computing. J. Electron. Inform. Technol. **34**(4), 812–817 (2012)
15. Wei, Z., Zhou, W., Ren, X., Wei, Q., et al.: A strategy-proof trust based decision mechanism for pervasive computing environments. Chin. J. Comput. **35**(5), 871–882 (2012)
16. Liu, W., Yin, L., Fang, B., Zhang, H.: A hierarchical trust model for the internet of things. Chin. J. Comput. **35**(5), 846–855 (2012)
17. Long, J., Liu, X., Yuan, X., Zhang, Z., et al.: A web services composition strategy based on trust reasoning and evolution. Chin. J. Comput. **35**(2), 298–314 (2012)
18. Qi, x, Jiyi, W., Guilin, W., Wenhao, L., et al.: Provably secure authentication protocol based on convertible proxy signcryption in cloud computing. Sci. China **42**(3), 303–313 (2012)
19. Jiyi, W., Ping, L., Pan, X., Zhuo, L.: Cloud computing: concept and platform. Telecommun. Sci. **33**(3), 59–66 (2012)
20. Li, Wenjuan, Ping, L., Li, J., Qiu, Q.: Cloud service discovery algorithm based on trust fuzzy comprehensive evaluation. ICIC Express Lett. Part B **3**(2), 1–6 (2012)
21. Li, W., Ping, L., Pan, X.: Trust model to enhance security and interoperability of cloud environment. In: Proceedings of the 1st International Conference on Cloud Computing (CloudCom'09), pp. 69–79. Springer, Berlin (2009)
22. Li, W., Ping, L., Qiu, Q., Zhang, Q.: Research on trust management strategies in cloud computing environment. J. Comput. Inform. Syst. **8**(4), 1757–1763 (2012)
23. Zhao, G.: Research on adaptive genetic algorithm based service workflow scheduling problem. Sun yat-sen university, Guangzhou (2010)

24. Yu, J., Tian, G., Cao, Y., et al.: A resource allocating algorithm in reliability. J. Comput. Res. Dev. **46**(2), 1821–1829 (2009)
25. Yuan, Y., Li, X., Wang, Q., et al.: Time optimization heuristics for scheduling budget_constrained grid workflows. J. Comput. Res. Dev. **46**(2), 194–201 (2009)
26. Li, J.: QoS Optimal, Grid Workflow Scheduling Algorithm. Guangxi University, Guilin (2009)
27. Wang, Y., Hu, C., Du, Z.: QoS-awared grid workflow schedule. J. Softw. **17**(11), 2341–2351 (2006)
28. Azzedin, F., Maheswaran, M.: Integrating trust into grid resource management systems. Proceedings of the 2002 International Conference on Parallel Processing, pp. 47–54. Vancouver, British Columbia (2002)
29. He, X., Sun, X., Gregor, V.L.: QoS guided Min-rain heuristic for grid task scheduling. J. Comput. Sci. Technol. **18**(4), 442–451 (2003)
30. Weng, C., Lu, X.: Heuristic scheduling for bag -of-tasks applications in combination with QoS in the computational grid. Future Gener. Comput. Syst. **21**(2), 271–280 (2005)
31. Buyya, R., Abramson, D., Giddy, J., et al.: Economic models for resource management and scheduling in grid computing. J. Concurr. Comput. **14**(13–15), 1507–1542 (2002)
32. Hum, P.M., Thompson, M.R.: Security implications of typical grid computing usage scenario, pp. 95–103. Proceedings of the IEEE HPDC, San Francisco, CA (2001)
33. Li K., He Y., Liu X.: Security-driven scheduling algorithms based on eigentrust in grid. In: Proceedings of the 6th International Conference of Parallel and Distributed Computing Applications and Technologies, Denver, USA, pp. 1068–1072 (2005)
34. Zhang, W., Fang, B., et al.: A trust-QoS enhanced grid service scheduling. chin. J. Comput. **29**(7), 1157–1166 (2006)
35. Yuan, L., Zeng, G., et al.: Dynamic level scheduling based on trust model in grid computing. Chin. J. Comput. **29**(7), 1217–1224 (2006)
36. Hu, Z., Juan, H.: Grid workflow scheduling algorithm under constraints of trust. Appl. Res. Comput. **27**(8), 2936–2938 (2010)
37. Hu, C., Wu, M., Liu, G.: QoS scheduling based on trust relationship in web service workflow. Chin. J. Comput. **32**(1), 42–53 (2009)
38. Hu, C., Wu, M., Liu, G., et al.: An approach to constructing web service workflow based on business spanning graph. J. Softw. **18**(8), 1870–1882 (2007)
39. Han, J.: Research on grid workflow scheduling algorithm under the constraints of trust. Zhongnan University, Changsha (2010)
40. Wang, X.: On the policy description and quantification model for trust management. National University of Defense Technology, Changsha (2009)
41. Li, W., Zhang, Q., Ping, L., Pan, X.: Cloud scheduling algorithm based on fuzzy clustering. J. Commun. **33**(3), 146–154 (2012)
42. Li, W., Pan, X., Zhang, Q., Ping, L.: A novel job scheduling model to enhance efficiency and overall user fairness of cloud computing environment. In: Proceedings of the 1st International Conference on Cloud Computing and Services Science (CLOSER2011), pp. 152–157. SCITePress, Noordwijkerhout (2011)
43. Bezdek, J.C.: Pattern Recognition with Fuzzy Objective Function Algorithms. Plenum Press, New York (1981)
44. NetLogo home page. http://ccl.northwestern.edu/netlogo
45. Calheiros, R. N., Ranjan1, R., De Rose, C. A. F., and Buyya, R.: "CloudSim: a novel framework for modeling and simulation of cloud computing infrastructures and services". http://www.chinacloud.cn/show.aspx?id=1316&cid=28
46. Buyya, R., Ranjan, R., and Calheiros, R.N.: "Modeling and simulation of scalable cloud computing environments and the CloudSim toolkit: challenges and opportunities". In: Proceedings of the 7th High Performance Computing and Simulation Conference (HPCS09), IEEE Computer Society, June 2009
47. Chung, J.Y., Shih, W.K., Jane, W.S., et al.: Scheduling periods jobs that allow imprecise results. IEEE Trans. Comput. **39**(9), 1156–1174 (1990)
48. Yu, A.C., Lin, K.J.: Scheduling parallelizable imprecise computations on multiprocessor. In: Proceedings of the 5th International Symposium on Parallel Processing. Anaheim, CA, USA, pp. 531–536 April 30–May 2 1991
49. Zhang, W., Liu, X., Yun, X., Zhang, H., et al.: Trust-driven job scheduling heuristics for computing grid. J. Commun. **27**(2), 73–79 (2006)
50. Zhao, C.: Research and Realization of Job Scheduling Algorithm in Cloud Environment. Beijing Jiaotong University, Beijing (2009)