Design and Optimization of Cloud-Oriented Workflow System

Lei Mao
School of Computer Science and Technology
School of Mineral Resource and Earth Science
China University of Mining and Technology, Xuzhou, China
Email: ml@cumt.edu.cn

Yongguo Yang
School of Mineral Resource and Earth Science
China University of Mining &Technology, Xuzhou, China
Email: ygyang88@cumt.edu.cn

Hui Xu
School of Computer science and Technology
School of Mineral Resource and Earth Science
China University of Mining & Technology, Xuzhou, China
Email: xuhui@cumt.edu.cn

***Abstract*—The paper firstly analyzes the characteristic of existing workflow applications and proposes cloud workflow layered model and design method based on the current major cloud computing trends. Considering the features on the cloud workflow system in distributed heterogeneous environment, the paper proposes a brief introduction to approach to cloud workflow system. Next, three deployment models of the cloud workflow applications are presented. Finally, the paper presents a new dynamic scheduling algorithm –Improved Dynamic Cloud Scheduling Algorithm(IDCSA) which takes consideration of the characteristics of cloud workflow and cloud resources.**

***Index Terms*—cloud workflow; layered model; style; distributed heterogeneous environment; deployment models; IDCSA**

I. INTRODUCTION

Workflow systems originated from office automation which started in 1970s to support the office information management for accomplishing simple business tasks . A workflow models a process as consisting of a series of steps, which simplifies the complexity of execution and management of applications. Today more companies are moving their workflow systems to online environments in order to streamline data hand-over and eliminate miscommunications that hinder flows at each step. This, in turn, leads to work optimization and better internal control. Moreover, more workflow systems support complex flows such as process splits, which offers alternate flows according to work data, and concurrent processing, which simultaneously sends the flow to multiple people.

Traditionally, a workflow management system is designed as an enterprise application which is deployed on company intranet, and its capability to support Internet-level mass usage is limited. These workflow systems are not suitable for open Internet environment. There are various workflow systems exist, but few of them provides public workflow service to users on the Internet.

Cloud services vary in the levels of abstraction and hence the type of service they present to application users. Infrastructure virtualization enables providers such as Amazon to offer virtual hardware for use in compute- and data-intensive workflow applications[1]. Platform-as-a-Service (PaaS) clouds expose a higher-level development and runtime environment for building and deploying workflow applications on cloud infrastructures. Such services may also expose domain-specific concepts for rapid-application development. Further up in the cloud stack are Software-as-a-Service providers who offer end users with standardized software solutions that could be integrated into existing workflows.

Cloud computing represents the future of the calculation of the development direction. Thus, it is possible to utilize cloud computing to address the problems of resource scalability and elasticity for managing large scale workflow applications. Therefore, the investigation of workflow systems based on cloud computing, namely cloud workflow systems is a timely issue and worthwhile for increasing efforts.

In this paper, we present a summary design of cloud workflow. We start by discussing five layers of cloud workflow system, compared to traditional workflow environments. Next, we give a brief introduction to

doi:10.4304/jsw.8.1.251-258

approach to cloud workflow system in order to highlight emphasis of change from workflow engines to clouds. Next, we present a study on three cloud workflow deployment models and the key challenges to realize them. Finally, we present Improved Dynamic Cloud Scheduling Algorithm (IDCSA). Simulative experiments show its performance improvement compared with other algorithm. Part V is the paper key content.

II. Layer Model of Cloud Workflow System

Different from traditional workflow environments, we split cloud workflow systems up into five layers: applications, software environments, software infrastructure, software kernel, and hardware (see figure 1). Obviously, at the bottom of the cloud stack is the hardware layer which is the actual physical components of the system. Many cloud computing offerings have built their system on subleasing the hardware in this layer as a service. At the top of the stack is the cloud workflow application layer, which is the interface of the cloud to the common computer users through web browsers and thin computing terminals.

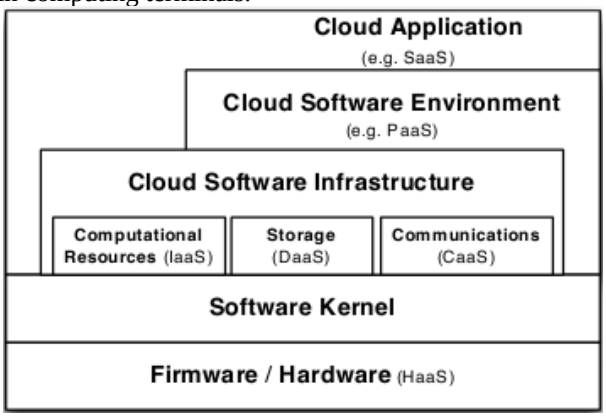


Figure 1. Layer model of of cloud workflow system.

A. Cloud Workflow Application Layer

The cloud workflow application layer is the most visible layer to the end-users of the cloud workflow. And in general, the users access the services provided by this layer through web-portals which are sometimes required to pay fees to use. This model has recently proven to be attractive to most users, as it alleviates the burden of software maintenance and the ongoing operation and support costs. Furthermore, it exports the computational work from the users' terminal to data centers where the cloud workflow applications are deployed. This in turn lessens the restrictions on the hardware requirements needed at the users' end, and allows them to obtain superb performance to some of their cpu-intensive and memory-intensive workloads without necessitating huge capital investments in their local machines.

B. Cloud Workflow Software Environment Layer

The second layer in the architecture is the cloud workflow software environment layer (also dubbed the software platform layer). The users of this layer are cloud workflow applications' developers, implementing their applications for and deploying them on the cloud. The providers of the cloud workflow software environments supply the developers with a programming-language-level environment with a set of well-defined APIs to facilitate the interaction between the environments and the cloud workflow applications, as well as to accelerate the deployment and support the scalability needed of those cloud workflow applications. The service provided by cloud systems in this layer is commonly referred to as Platform as a Service (PaaS). One example of systems in this category is Google's App Engine, which provides a python runtime environment and APIs for applications to interact with Google's cloud runtime environment[2]. Another example is Sales Force Apex language that allows the developers of the cloud applications to design, along with their applications' logic, their page layout, sequence of operations, and customer reports. [3]

C. Cloud Workflow Software Infrastructure Layer

The cloud workflow software infrastructure layer provides fundamental resources to other higher-level layers, which in turn can be used to construct new cloud workflow software environments or cloud workflow applications. The architecture reflects the fact that the two highest levels in the cloud stack can bypass the cloud workflow infrastructure layer in building their system. Although this bypass can enhance the efficiency of the system, it comes at the cost of simplicity and development efforts.

D. Software Kernel

This cloud workflow layer provides the basic software management for the physical servers that compose the cloud. Software kernels at this level can be implemented as an OS kernel, hypervisor, virtual machine monitor and/or clustering middleware. Customarily, grid computing applications were deployed and run on this layer on several interconnected clusters of machines. However, due to the absence of a virtualization abstraction in grid computing, jobs were closely tied to the actual hardware infrastructure and providing migration, check pointing and load balancing to the applications at this level was always a complicated task.

E. Hardware and Firmware

The bottom layer of the cloud stack in the architecture is the actual physical hardware and switches that form the backbone of the cloud workflow. In this regard, users of this layer of the cloud workflow are normally big enterprises with huge IT requirements in need of subleasing Hardware as a Service (HaaS). For that, the HaaS provider operates, manages and upgrades the hardware on behalf of its consumers, for the life-time of the sublease. This model is advantageous to the enterprise users, since they do not need to invest in building and managing data centers. Meanwhile, HaaS providers have the technical expertise as well as the cost-effective infrastructure to host the systems. One of the early examples HaaS is Morgan Stanley's sublease contract with IBM in 2004[4]. SLAs in this model are more strict, since enterprise users have predefined business

workloads whose characteristics impose strict performance requirements. The margin benefit for HaaS providers materialize from the economy of scale of building huge data centers infrastructures with gigantic floor space, power, cooling costs as well as operation and management expertise.

III. Approach to Cloud Workflow System

The primary benefit of moving to clouds is application scalability. Unlike grids, scalability of cloud resources allows real-time provisioning of resources to meet application requirements at runtime or prior to execution. The elastic nature of clouds facilitates changing of resource quantities and characteristics to vary at runtime, thus dynamically scaling up when there is a greater need for additional resources and scaling down when the demand is low. This enables workflow management systems (WfMSs) to readily meet quality-of-service (QoS) requirements of applications, as opposed to the traditional approach that required advance reservation of resources in global multi-user grid environments. With most cloud computing services coming from large commercial organizations, service-level agreements (SLAs) have been an important concern to both the service providers and consumers. Due to competitions within emerging service providers, greater care is being taken in designing SLAs that seek to offer better QoS guarantees to customers and clear terms for compensation in the event of violation. This allows workflow management systems to provide better end-to-end guarantees when meeting the service requirements of users by mapping them to service providers based on characteristics of SLAs. Economically motivated, commercial cloud providers strive to provide better services guarantees compared to grid service providers. Cloud providers also take advantage of economies of scale, providing compute, storage, and bandwidth resources at substantially lower costs. Thus utilizing public cloud services could be economical and a cheaper alternative (or add-on) to the more expensive dedicated resources. One of the benefits of using virtualized resources for workflow execution, as opposed to having direct access to the physical machine, is the educed need for securing the physical resource from malicious code using techniques such as sandboxing[5]. However, the long-term effect of using virtualized resources in clouds that effectively share a "slice" of the physical machine, as opposed to using dedicated resources for high-performance applications, is an interesting research question.

Traditional WfMSs were designed with a centralized architecture and were thus tied to a single computer. In distributed heterogeneous environment, moving workflow engines to clouds requires architectural changes, method of scale activity node and integration of cloud management tools.

A. Architectural Changes

Most components of a WfMS can be separated from the core engine so that they can be executed on different cloud services. Each separated component could communicate with a centralized or replicated workflow engine using events. The manager is responsible for coordinating the distribution of load to its subcomponents, such as the Web server, persistence, monitoring units, and so forth.

In new WfMS, we should separate the components that form the architecture into the following: user interface, core, and plug-ins. The user interface can now be coupled with a Web server running on a "large" instance of cloud that can handle increasing number of users[6]. The Web request from users accessing the WfMS via a portal is thus offloaded to a different set of resources.

Similarly, the core and plug-in components can be hosted on different types of instances separately. Depending on the size of the workload from users, these components could be migrated or replicated to other resources, or reinforced with additional resources to satisfy the increased load. Thus, employing distributed modules of the WfMS on the basis of application requirements helps scale the architecture.

B. Method of Scale Activity Node

Heavy workflow of traditional waterfall approaches with smallest detail will slow down the use of cloud computing. So it is necessary to separate main workflow from details of mechanism required to scale any activity node. Thus we must have efficient way of storing this information

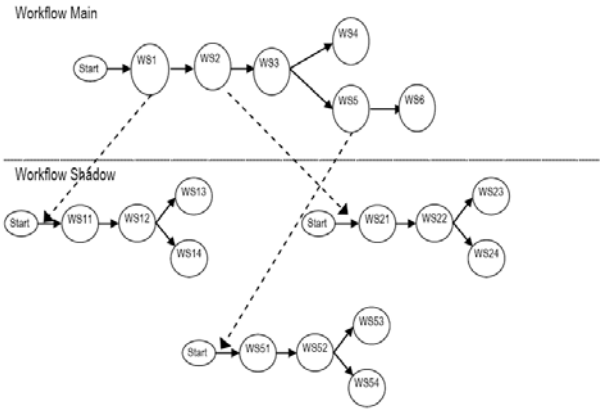


Figure 2. A example of workflow main and workflow shadow

In Figure 2, we see two "different" workflows, workflow main that has the cloud structure with each web service as an activity node, workflow shadow that has sub-workflows for other options for each activity nodes. Of course we can also divide it into workflow online and offline. The former runs and executes at a particular time and the latter is a kind of workflow in passive state waiting for an event to trigger it.

The following table I gives a detailed description of the node activities.

TABLE I.

DESCRIPTION OF THE NODE ACTIVITIES

Parameters	Description
Activity Node Name	/* Unique Activity node Name*/
Description	/* Description of Activity */
Type	/* Service or Application etc*/
State	/* Online, Offline, or Needs change*/
Constraints	/* Time, Execution Cost */
Input Event	/*Event to trigger the activity node */
Output Event	/*Event triggered by activity node */
Scalability Options	/*Scalability activities as a workflow */ (When and How)
Compressibility Options	/*Compressibility activities as a workflow*/ (When and How)

C. Integration of Cloud Management Tools

As the WfMS is broken down into components to be hosted across multiple cloud resources, we need a mechanism to access, transfer, and store data and enable and monitor executions that can utilize this approach of scalable distribution of components.

The cloud service provider may provide APIs and tools for discovering the VM instances that are associated to a user's account. Because various types of instances can be dynamically created, their characteristics such as CPU capacity and amount of available memory are a part of the cloud service provider's specifications. Similarly, for data storage and access, a cloud may provide data sharing, data movement, and access rights management capabilities to user's applications. Cloud measurement tools may be in place to account for the amount of data and computing power used, so that users are charged on the pay-per-use basis. A WfMS now needs to access these tools to discover and characterize the resources available in the cloud. It also needs to interpret the access rights (e.g., access control lists provided by Amazon), use the data movement APIs, and share mechanisms between VMs to fully utilize the benefits of moving to clouds. In other words, traditional catalog services such as the Globus Monitoring and Discovery Service (MDS) Replica Location Services, Storage Resource Brokers, Network Weather Service, and so on could be easily replaced by more user-friendly and scalable tools and APIs associated with a cloud service provider[7].

A WfMS implements scheduling policies to assign tasks to resources based on applications' objectives. This task-resource mapping is dependent on several factors: compute resource capacity, application requirements, user's QoS, and so forth. Based on these objectives, a WfMS could also direct a VM provisioning system to consolidate data center loads by migrating VMs so that it could make scheduling decisions based on locality of data and compute resources[8].

Most cloud providers also offer services and APIs for tracking resource usage and the costs incurred. This can complement workflow systems that support budget-based scheduling by utilizing real-time data on the resources used[9], the duration, and the expenditure. This information can be used both for making scheduling decisions on subsequent jobs and for billing the user at the completion of the workflow application.

Ⅳ. CLOUD WORKFLOW DEPLOYMENT MODELS

There are three main types of cloud workflow deployment models - public, private and hybrid cloud workflow.

A. Public Cloud Workflow

Public cloud workflow is the most common type of cloud workflow. This is where multiple customers can access web applications and services over the internet. Each individual customer has their own resources which are dynamically provisioned by a third party vendor. This third party vendor hosts the cloud for multiple customers from multiple data centers (see figure 3), manages all the security and provides the hardware and infrastructure for the cloud to operate. The customer has no control or insight into how the cloud is managed or what infrastructure is available.

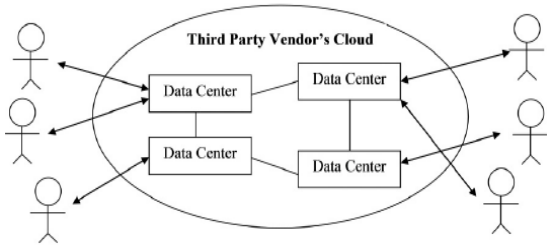


Figure 3. Public cloud workflow deployment model.

B. Private Cloud Workflow

Private cloud workflow emulates the concept of cloud computing on a private network. They allow users to have the benefits of cloud workflow without some of the pitfalls. Private cloud workflow grants complete control over how data is managed and what security measures are in place. This can lead to users having more confidence and control. The major issue with this deployment model is that the users have large expenditures as they have to buy the infrastructure to run the cloud workflow and also have to manage the cloud workflow themselves.

C. Hybrid Cloud Workflow

Hybrid cloud workflow incorporates both public and private cloud (see figure 4) within the same network. It allows the organizations to benefit from both deployment models. For example, an organization could hold sensitive information on their private cloud workflow and use the public cloud workflow for handling large traffic and demanding situations.

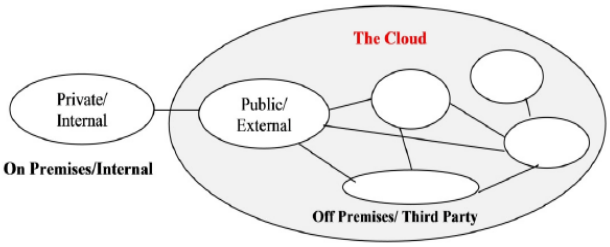


Figure 4. Hybrid cloud workflow deployment model.

Ⅴ.IMPROVED DYNAMIC CLOUD SCHEDULING ALGORITHM

DAG(directed acyclic graph) is a common method to describe the workflow. Many researchers used to use linear program, genetic algorithm, simulate anneal arithmetic, and hybrid algorithm based on PSO et al. to solve the scheduling problem of the workflow described by DAG. But when the problem is in large scale, the algorithm of time is costly. Structural type heuristic algorithm is used by many researchers to solve the optimization of the time cost based on workflow scheduling problem to ensure the acceptable solution in reasonable operation time. Considering two quality parameters(time and cost of the service resources), based on deadline/cost constraint, R.Buyya put forward a task scheduling heuristic algorithm of three optimization strategy respectively according to time, cost and time variants. And in the literature [10] he finally put forward a basis for the optimization tactics of the time-cost. These algorithm did not consider the workflow between tasks control dependence and the task of the relationship of the optional services for scheduling performance influence. So they can only get the shortest time (and also the highest cost) , the cost minimum algorithm (and also the longest time) and several other simple scheduling scheme. Although to some extent it solves the time cost optimization problem of the existing cloud workflow, but it is not universal.

Based on network diagram decomposition, the literature [11] used the column generation technology to provide a method to realize the bounds solving task scheduling. Jia Yu[12] and Yuan[13] put forward DTL(Deadline Top Level)algorithm and DBL(Deadline Bottom Level) algorithm. According to the model of workflow to hierarchical structure, they averagely assigned deadline to each layer deadline and eventually get global approximate optimal solution through the layers of the activities of local cost optimization. The literature [14] further put forward the concept of the string protocol and realized the optimal time to collect and use by a group of string protocol. The two types of algorithm focus on the network diagram analysis of workflow applications and don't take into consideration specific node service pool. Because the width of the layered and the cost of the interval optimization is tiny and it does not consider the task of optional service the link between algorithm, there is a large space of improvement in the performance of algorithm.

According to priority algorithm and optimization algorithm of deadline is insufficient, taking into account each node of the pool service, the paper presents a new dynamic scheduling algorithm–Improved Dynamic Cloud Scheduling Algorithm (IDCSA) in which time and cost will be bound together. Because of a variety of strategies in dynamic support and relative protection for acquiring resource and organization information relative protection, based on the macro understanding about resources situation involved with cloud workflow, IDCSA can guarantee the execution of scheduling of whole cloud workflow.

A. The Definition of IDCSA's Parameters

Before IDCSA is used, its parameters, including workflow critical path, key factor, dynamic factor, priority factor, must be defined.

AA. Workflow critical path

According to static calculation of the cloud workflow task instances, we can get the biggest completion time of the whole workflow instance. Considering the workflow instances in which there are some uncertain factors, such as selecting, etc, we use $TC=\{T_j\}$ as the Migration set composed of the critical path and TT as expected time. So we give a key factor to key tasks on workflow critical path. Considering the path of the choosing condition, critical path may be changed in the implementation process. Therefore if the condition path is the critical path, the we need to find time critical path for backup.

AB. Key Factor(KF)

$$KF_i = \frac{TE_i}{TT} \qquad (1)$$

where $T_i \in TC$;K is a constant value; E_i is the tuning parameter which regards that the key factor of other tasks is 0 and means the influence on running time of the task TC may be changed during the implementation, so the key factor of the tasks may be changed also. Thus we can adjust the value of KF and give it to 10 levels.

AC. Dynamic Factor (DF)

DF means the start time of the task execution and the expectant start time.

$$DF_i = \frac{T_i - TB_i - TE_i}{TT} \qquad (2)$$

If the value of DF is positive, it shows that the time of task is very rich. And if he value of DF is negative, it shows that the task has been postponed. In order to distinguishing different situation, we can give the value of DF some levels

-10,-9,…, −2, −1, 0, 1, 2, …, 9, 10

Through parameter K we can make some adjustment and processing. For example, it can be defined as follows

$$LDF_i = \begin{cases} DF_i > 0.1 : 10 \\ 0.09 < DF_i \leqslant 0.1 : 9 \\ \cdots\cdots \\ 0 < DF_i \leqslant 0.01 : 1 \\ 0 : 0 \\ -0.01 < DF_i \leqslant 0 : -1 \\ \cdots\cdots \\ -0.1 < DF_i \leqslant -0.09 : -9 \\ DF_i \leqslant -0.1 : -10 \end{cases} \qquad (3)$$

AD. Priority Factor (PF)

According to a given task priority(such as service quality) ,we can define PF of the task. The priority factor of task T_i is PF whose value depends on priority level of workflow. We can give the value of PF ten levels:0,1,2, ⋯,9.

B. The Description of IDCSA

The search procedures of the proposed IDCSA are detailed as follows.

Step1. Task selection: Initialize a cloud workflow process; define the value of each parameter; finally output real-time dynamic ready tasks queue of cloud workflow (RTQ). In this step, ready tasks queue always be changed over time and the change is related to scheduling and resources.

Thus some of these factors need to be recount. According to relationship to model of cloud workflow, it should establish the corresponding dynamic ready distribution queue to the grid workflow instance.

Step2. task scheduling: define $RTQ=\{T_1,T_2, \cdots ,T_n\}$,organization resources pooling $OR=\{OR_1,OR_2,\cdots,OR_m\}$.The question about organization resources assigned to the task is actually looking for a 1:m mapping from RTQ to OR and then allocating the appropriate to the appropriate. During allocation, the main consideration is that does OR whether meet the resource requirements, organization role needs and Organization role needs and service quality.

Step3. Resource Selection: input task T_j and resources pooling $OR=\{OR_1,OR_2,\cdots,OR_m\}$;output resources queue ORT meeting task T_j in which the queue is classified.

Step4. Resource allocation: According to various parameters of the mission and the quality of resources, resources are allocated dynamically. The algorithm first takes into consideration the requirements of the quality because of parameters combination. A quality requirement factor(RF) can be composed of three factors(LDF,LKF and PF).

$$RF=PF*(11-LDF)(LKF+1). \quad (4)$$

So the distribution thought is as follows: the more value RF_j of task T_j. , the better the quality resources is allocated from resource queue ORT_j.

Step5. Monitoring feedback: Owing to dynamic characteristics of cloud Resources, in the implementation process cloud resources organization may exit or join dynamically in the implementation. So it maybe bring some of the effects to scheduling. We use the feedback mechanism to solve the missions which are not implemented successfully or seriously overtime.

Step6. Check if the terminal condition is satisfied: If the terminal condition has not been satisfied, go to Step2; otherwise, the optimization process ends.

When the cloud workflow task is distributed to implementation for some members within the organization, according to their own resources scheduling strategy, the organization begins to schedule processing. The scheduling process of algorithm is detailed as Figure 5.

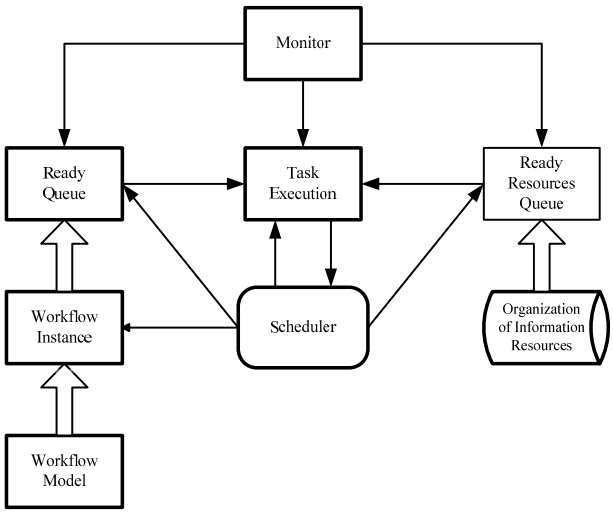
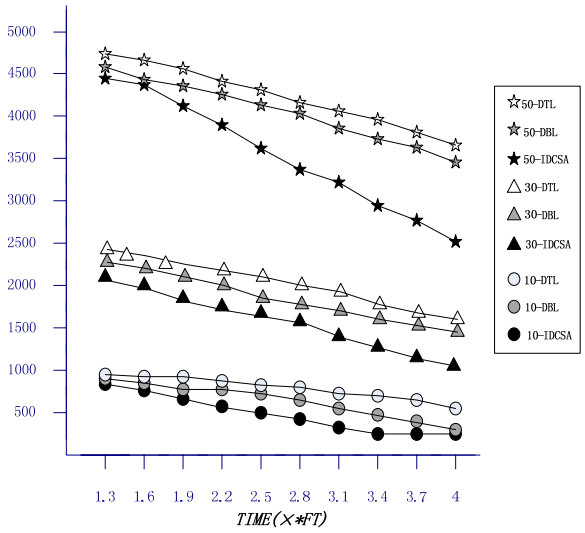


Figure 5. Scheduling Process of Algorithm.

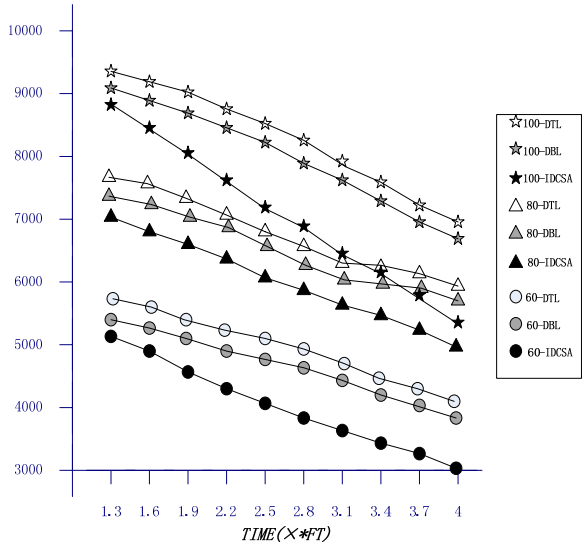
For circumstance of multiple workflow instance scheduling, there are maybe the same cloud workflow model of multiple instances or the different cloud workflow model of multiple instances. Considering scheduling, cloud workflow engine is multi-tasking. The different instances are no related and the different tasks are not linked. So the main consideration of algorithm is the competition of the resources and the algorithm needs to deal with different instances of priority in concern with resource allocation and scheduling

C. Simulation and Performance Evaluation

In order to assess the performance of the algorithm, we have simulation tests to different DAG workflow application and compare IDCSA to DTL and DBL. Simulation environments are as follows: operating system is Windows 7.The program runs on PC of core 2 duo, 1.8GHZ, 2G RAM. The programming language of algorithm is java. The DAG of Workflow instance is automatically generated and the automatic generator of DAG needs to set up |V| nodes. The service quantity of each node service pool is a random number of 15 seeds. The service operating time is a increasing random number of 10 seeds. The cost is a diminishing random number of 10 seeds. The expense is the time discrete strictly decreasing function. All random numbers are evenly distributed. In the test there are 10 application nodes which set the values in {10,20,30,40,50,60,70,80,90,100}. Each application set s 10 different instances of deadline. These instances of deadline respectively promote an l rate of more than 30 percent increase in contrast to the minimum completion time FT. Each experiment result of application is all it's average of the instances. Figure 6 shows cost comparison of different scheduling algorithms with different tasks and deadlines.



(a) 10/30/50 Tasks



(b) 60/80/100 Tasks

Figure 6. Performance improvement of IDCSA under different deadlines.

A_k is the cost of scheduling algorithm A of which number of nodes is K. And .$IDCSA_k$ is the cost of scheduling algorithm IDCSA of which number of nodes is K.K∈{10,20,30,40,50,60,70,80,90,100}.

$$I_A = \frac{1}{10} * \frac{\sum_K (IDCSA_K - A_K)}{A_K} \quad (5)$$

TABLE II.

PERFORMANCE IMPROVEMENT OF IDCSA

	DTL	DBL
1.3*FT	0.049	0.029
1.6*FT	0.063	0.044
1.9*FT	0.083	0.055
2.2*FT	0.114	0.087
2.5*FT	0.136	0.107
2.8*FT	0.149	0.124
3.1*FT	0.171	0.146
3.4*FT	0.192	0.165
3.7*FT	0.219	0.191
4*FT	0.238	0.212

Aiming at the cloud workflow scheduling with the objective of time-cost optimization, the cloud tasks scheduling phases and policy are analyzed. The cloud workflow dynamic scheduling algorithm with multi-policy is presented, which take consideration of the characteristics of cloud workflow and grid resources. The scheduling of cloud workflow tasks can be effectively implemented by the Key Factor, Dynamic Factor and Prior Factor.

VI. CONCLUSION

To summarize, we have presented five-layer model of cloud workflow system. We discussed the limitations of existing workflow management systems and proposed changes that need to be incorporated when moving to clouds. We also described three cloud workflow deployment models. Finally, as the focus of this paper, we present IDCSA which takes consideration of the characteristics of cloud workflow and cloud resources.

Based on our experience in using cloud services, we conclude that large applications can certainly benefit by using cloud resources. The key benefits are in terms of decreased runtime, on-demand resource provisioning, and ease of resource management. In the near future, workflow management matters because much of the benefits of cloud computing comes from the speed and ease with which IT resources can be created and put into production.

ACKNOWLEDGE

This work was supported by the National Natural Science Foundation of China (No. 40972207), the National Science and Technology Major Projects (No. 2011ZX05034-005) and the PAPD of Jiangsu Higher Education Institutions. These supports are gratefully acknowledged.

REFERENCES

[1] R. Buyya, S. Pandey,and C. Vecchiola, *Cloudbus toolkit for market-oriented cloud computing, in Proceedings of the 1st International Conference on Cloud Computing* (CloudCom 2009,Springer,Germany),Beijing, China, December 14,2011,pp. 78-84,.

[2] C. Hoffa et al., *On the Use of Cloud Computing for Scientific Workflows*, 3rd International Workshop on Scientific Workflows and Business Workflow Standards in e-Science (SWBES '09), 2009，pp. 241-245

[3] S.Pandey, W.Voorsluys, M. Rahman, R.Buyya,J. Dobson,and K. Chiu, “A grid workflow environment for brain imaging analysis on distributed systems, Con-

currency and Computation": *Practice and Experience* , 21 (16), pp. 2118-2139 ,2009
[4] J. Yu et al., "A Taxonomy of Workflow Management Systems for Grid Computing", *Journal of Grid Computing* , vol. 3, pp. 121-125, 2011
[5] E. Deelman et al., "Pegasus: A framework for mapping complex scientific workflows onto distributed systems", *Scientific Programming* , vol. 13, pp. 219-237, 2011
[6] J.Yu and R.Buyya, "Scheduling scientific workflow applications with deadline and budget constraints using genetic algorithms", *Scientific Programming Journal* ,Vol. 14 ,pp.217-230 , 2006
[7] Google appEngine, http://code.google.com/appengine/, November 2009.
[8] D. Nurmi et al., *The Eucalyptus Open-source Cloud-computing System*, IEEE International Symposium on Cluster Computing and the Grid (CCGrid '09) , Beijing, China, April,2009,pp. 654-661,
[9] F. Chang,J. Dean,S. Ghemawat, W. C. Hsieh, D. A. Wallach, M.Burrows,T.Chandra, A.Fikes,and R.E. Gruber,Bigtable, *A distributed storage system for structured data*, Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation, Berkeley, CA, USENIX Association,2006, pp.15-17.
[10] S. Ghemawat, H. Gobioff,and S. T. Leung,"The google file system", *SIGOPS Operating System Review* , Vol.37 ,pp.29 -43,2007
[11] Akkan C, Drexl A, and Kimms A,"Network decomposition-based benchmark results for the discrete time-cost tradeoff problem". *European Journal of Operational Research.* 165(2).pp. 339-358,2009
[12] Jia Yu, Rajkumar Buyya, and Chen Khong Tham. *Cost-based scheduling of workflow applications on utility grids*. The 1st IEEE Int'l Conf on e-Science and Grid Computing, Melbourne,Australia, IEEE Press. 2007,pp.140-147.
[13] Yuan Ying, Chun LI, and Qian Zhang, "Bottom Level Based Heuristic for Workflow Scheduling in Grids", *Chinese Journal of Computer*. 31, (2),pp.282-290, 2011
[14] Yuan Ying, Li Xiaoping, and Wang Qian , "Cost Optimization Heuristics for Grid Workflows Scheduling Based on Serial Reduction", *Journal of Computer Research and Development.* 45 (2),pp.246-253, 2011

Lei Mao is currently a Ph.D candidate at China University of Mining and Technology, China. He received his MS degree in Computer application Technology from China University of Mining and Technology in 2004, and his BS degree in Computer Science from China University of Mining and Technology in 2000. He is currently a lecturer at School of Computer Science and Technology, China University of Mining and Technology. His research interests include cloud computing, workflow system and parallel processing et al.

Yongguo Yang, born in 1962, Ph.D. He is a professor at school of Mineral Resource and Earth Science and a director of Geo-Information Science Institute in CUMT. His research interests are mathematical geology and GIS applications. He has published 3 books, and more than 40 research papers in journals and international conferences. Now he preside the National Natural Science Foundation of China (No. 40972207), the National Science and Technology Major Projects (No. 2011ZX05034-005) and the PAPD of Jiangsu Higher Education Institutions.

Hui Xu is current a Ph.D .candidate at China University of Mining and Technology(CUMT), China. She received her MS degree in Computer Application Technology from CUMT in 2005, and her BS degree in Computer Science from CUMT in 2002. She is currently a lecture at school of Computer Science and Technology, CUMT. Her research interest is computation intelligence and coalbed methane et al.