OmniaScience

Journal of Industrial Engineering and Management
JIEM, 2015 – 8(1): 217-232 – Online ISSN: 2013-0953 – Print ISSN: 2013-8423
http://dx.doi.org/10.3926/jiem.1363

Resource Scheduling of Workflow Multi-instance Migration Based on the Shuffled Leapfrog Algorithm

Yang Mingshun, Gao Xinqin, Cao Yuan, Liu Yong, Li Yan

Faculty of Mechanical and Precision Instrument Engineering, Xi'an University of Technology (China)

yangmingshun@xaut.edu.cn, gaoxinqin@xaut.edu.cn, nocc@163.com, liuyong@xaut.edu.cn, jyxy-ly@xaut.edu.cn

Received: December 2014
Accepted: February 2015

Abstract:

Purpose: When the workflow changed, resource scheduling optimization in the process of the current running instance migration has become a hot issue in current workflow flexible research; purpose of the article is to investigate the resource scheduling problem of workflow multi-instance migration.

Design/methodology/approach: The time and cost relationships between activities and resources in workflow instance migration process are analyzed and a resource scheduling optimization model in the process of workflow instance migration is set up; Research is performed on resource scheduling optimization in workflow multi-instance migration, leapfrog algorithm is adopted to obtain the optimal resource scheduling scheme. An example is given to verify the validity of the model and the algorithm.

Findings: Under the constraints of resource cost and quantity, an optimal resource scheduling scheme for workflow migration is found, ensuring a minimal running time and optimal cost.

Originality/value: A mathematical model for resource scheduling of workflow multi-instance migration is built and the shuffled leapfrog algorithm is designed to solve the model.

Keywords: multi-instance migration, resource scheduling, leapfrog algorithm

1. Introduction

The sustainable development of business enterprise is threatened by the contention between rapidly-increasing business volume and limited enterprise resources. As the core technology of process modeling and management, workflow technology is of key importance for improving information level, production and operations management, operation efficiency, response ability to the external environment change and market competitiveness of manufacturing enterprises (Xu, 2014). Workflow resource scheduling is the primary concern of workflow management systems. To determine how resources can be most appropriately used to execute workflow instances, thereby improving the execution efficiency of business processes has been the primary focus.

Due to increasing market competitiveness and the complex and ever-changing demands of enterprise, uncertainty and variability have become significant features of business enterprise (Feifan & Xuanxi, 2006). Workflow instance migration can effectively solve the problem of workflow dynamic change. As the workflow changes, the running instance will choose an appropriate migration strategy (direct migration, rollback migration, or no migration) according to the very status (Gao, Xu, Wang, Li, Yang & Liu, 2013). During the process of workflow migration, the workflow process model will change; the resources relevant to the original process model will also change, such as resource rebuilding or canceling, which makes the resource scheduling optimization in the workflow migration process very important. How to migrate currently running instances and schedule resources during instance migration process has become widely debated issue in workflow flexibility research.

Aiming to resolve the grid workflow scheduling problem, Sucha Smanchat et al. proposed a scheduling algorithm for a multi-parameter sweep workflow instance based on resource competition (Smanchat, Indrawan & Ling, 2011). Rizos Sakellariou et al. considered resource allocation problems to be a single activity instance of the workflow and set the earliest completion time for a certain activity instance as the goal of their resource scheduling method (Sakellariou, Zbao, Tsiakkouri & Dikaiako, 2007). R. Buyya analyzed the relationship between the overall deadline of a workflow instance and the load of an activity instance in order to estimate the deadline for each activity's running time. The workflow instance resource scheduling problem has been developed into multiple scheduling problems (Yu, Buyya & Tham, 2005). G. B. Tramontina et al. adopted a variety of allocation rules (First In First Out, Earliest Due Time, Service In Random Order, and Shortest Processing Time) in order to schedule resources among multiple workflow instances (Tramontina & Wainer, 2005).

Li Shengwen et al. established a workflow resource scheduling model based on fuzzy theory; when the process was running, the fuzzy information from the relevant environment was analyzed (Shengwen & Junfang, 2010). This model was successful in the optimal resource scheduling of workflow instances, but the weights of the resources and self-adaption were not

considered. Zhang Lijun solved the problem of dynamic task allocation using sub-processes; although the design and execution of the sub-processes were given in this method, it paid less attention to resource task scheduling (Lijun, Jun & Tao, 2009). Liu Subo established a resource-task allocation model, and designed a process-scheduling decision system based on workflow management, resolving the problems arising from resource optimization scheduling and exception handling (Subo, Jianchong & Haizhu, 2011). Deng Tieqing et al. studied the personal worklist scheduling problem in an environment of dynamic execution of workflow instances, and proposed a personal worklist resource scheduling algorithm based on a genetic algorithm; this method studied resource scheduling of multiple activity instances in a single process, but not in resources scheduling of multiple process instances (Deng, Ren & Liu, 2012). Yang Mingshun et al. established a workflow multi-instance resource optimization model based on queuing theory and used the simulated annealing intelligent algorithm to solve the model (Yang, Han, Gao & Liu, 2012). This model solved the resource optimization problem with the constraints of resource cost and state as the instances were running, but the resolution lacked the consideration of resource competition and its advantages and disadvantages.

2. Resource Scheduling Modeling in Instance Migration Process

2.1. Workflow Instance Migration Process Analysis

The workflow instance migration process primarily includes modeling of dynamic change of workflow, workflow instance migration, and resource scheduling optimization.

(1) Modeling of dynamic change of workflow refers to build a new or composite workflow process model of the changed workflow process based on the original workflow model. Firstly, it is required to analyze the change style of the workflow, then the new/composite workflow process models corresponding to the original ones are built, finally the structures of the newly built models are verified to determine whether any conflict or error exists. Only no conflict or error existing in the workflow structure, the new model is reasonable, and the instances run in the original model can migrate.

(2) When changes occur in the workflow, based on the built new/composite workflow process model, instances run in the old workflow model are required to migrate in real time. The steps of the instance migrating includes identifying the workflow regions, sequencing the workflow regions and selecting the migrating strategies. The workflow regions identifying includes identifying the regions of the current nodes running and the dynamic changing regions. Then using the workflow regions sequencing rules, the current running regions and all changing regions are sequenced. Finally, based on the sequencing results, the appropriate migrating strategies are selected to finish migrating of the current running instances.

(3) It is well-known that the successful running of the workflow instances cannot do with appropriate resources, especially in the process of workflow instances migration. Only with the supports of appropriate resources, can the instances migrating be successfully realized. The purpose of workflow resource scheduling is to allocate the right activities to the right activities in right time with right sequence to ensure the every activity of the workflow instances can be executed by the appropriate resource in right time, thus the whole business process can be finished successfully. The resource scheduling results will directly affect the work efficiency, flexibility and service quality of the workflow management system.

The framework of the workflow instance migration process is shown in Figure 1.

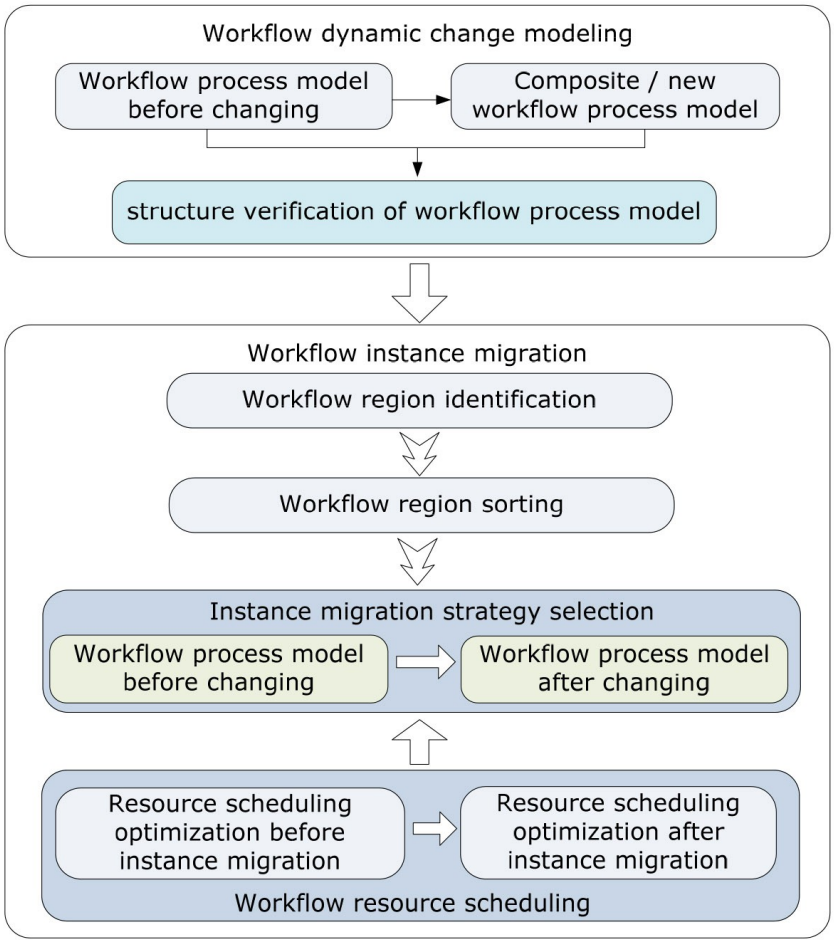

Figure 1. Process framework of the workflow instance migration

The resource scheduling includes task allocation and task sequencing. Task allocation refers to allocate the appropriate resources to the work items, which is the core problem of workflow resource scheduling. While task sequencing refers to priority ordering of the multi tasks allocated to a same resource.

In the workflow instance migration process, due to uncertain factors such as resource changes and resource release, the dynamic resource scheduling optimization should be applied to the migration instance so that the workflow system can achieve global optimization of resource utilization. By analyzing the relationship between the activities and resources in the workflow instance migration process, certain factors, including the shortest overall activity execution time, the minimum overall activity lag time, the minimum number of activities lagged and the minimum costs that the resource consumed were used as the optimization goals to establish the mathematical model of workflow resource scheduling optimization.

2.2. Resource Scheduling Parameters

(1) Activity *i's* lag time (Δt_i) represents the time difference between the expected activity completion time and the actual completion time.

$$\Delta t_i = te_i - tf_i \tag{1}$$

where te_i is the expected completion time of activity *i* and tf_i is the actual completion time of the activity. The expected completion time of activity *i* is the time expected to meet the time requirements or obtain certain time performance. The expected completion time could be set by users or generated by the system.

(2) Activity *i*'s actual completion time (tf_i) represents activity *i's* actual starting time plus its execution time.

$$tf_i = ts_i + t_{ij} \tag{2}$$

where ts_i is the starting time, t_{ij} is the time resource *j* spends executing activity *i*.

(3) The execution time (t_{ij}) of resource *j*, or the time resource *j* spends executing activity *i*, represents the ratio of the workload of activity *i* to the executive ability of resource *j*.

$$t_{ij} = \alpha \cdot \frac{W_i}{A_{ij}} \tag{3}$$

where W_i is the workload of activity *i*, A_{ij} is resource *j*'s ability to execute the activity, and α is a correction coefficient.

(4) The consumption cost (C_{ij}) of resource *j* in the execution of activity *i* represents the average consumption cost of resource *j* per unit of time multiplied by the activity *i*'s running time.

$$C_{ij} = c_j \cdot t_{ij} \tag{4}$$

where c_j is the average consumption cost of resource j per unit of time, and t_{ij} is the time that resource j spends executing the activity.

(5) The available time of resource j (tr_j) includes the current time, the idle time of resource i, and the execution time already allocated to resource j.

$$tr_j = \begin{cases} tc + \sum_{k=1}^{\lambda} tp_{jk}, tv_j < tc \\ ts_j + \sum_{k=1}^{\lambda} tp_{jk}, tv_j \geq tc \end{cases} \tag{5}$$

where tv_j is the idle time of resource j, tc is the current time, $\sum_{k=1}^{\lambda} tp_{jk}$ is the overall execution time already allocated to resource j, and λ is the number of tasks already been allocated to resource j.

(6) In case one resource serves several activities, virtual resources need to be introduced in order to obtain an optimal resolution of the activity and the resource sorting problem. Assuming that resource j needs to participate in β executing activities, β-1 virtual resources of resource j need to be added. The physical and virtual resources are expressed as $j.k$ (k = 0, …, s - 1). Then, equation (5) could be modified as the following:

$$tr_{j.k} = \begin{cases} tc + \sum_{q=0}^{k-1} tp_{jq}, tv_j < tc \\ ts_j + \sum_{q=0}^{k-1} tp_{jq}, tv_j \geq tc \end{cases} \tag{6}$$

Where tp_{jq} is the allocated time that resource j spends executing activity q.

(7) The actual starting time of the activity depends on the available time of the resource.

$$ts_i = tr_{j.k} \tag{7}$$

Where $tr_{j.k}$ is the available time of resource $j.k$, and $j.k$ is the resource that executes activity i.

The start time of an activity refers to the moment that the instance activity i begins to be executed with some resources and can be represented with tb_i.

Free time of a resource refers to a moment after which the resource *j* will not participant in execution of any activity and can be represented with tv_j. Available time of a resource refers to the moment that the resource *j* can participant in the execution of a new activity and can be represented with tv_j.

When an activity needs multi resources to be executed, which means a teamwork. Multi resource execution styles are required to be taken into account. There are mainly four styles: different time and place, same time and place, same time and different place, different time and same place. In the process of workflow instance running, only the time factor of resource executing is considered, thus in the paper the style of same time and place of the resource teamwork is mainly considered.

(8) The execution time ($t_{i(wt_\theta)}$) of activity *i* by team wt_θ depends on the completion time of the resource that spends the longest time executing the activity.

$$t_{i(wt_\theta)} = Max\left(\gamma_{(wt\theta)h} \cdot \frac{W_{i(wt_\theta)_h}}{A_{i(wt_\theta)_h}}\right), h = 1,\ldots,|wt_\theta| \tag{8}$$

where $\gamma_{(wt\theta)h}$ is the modified coefficient of the demanding time that resource member *h* of team wt_θ spends executing activity *i*, $w_{i(wt_\theta)_h}$ is the workload executed by resource $(wt_\theta)_h$, and $A_{i(wt_\theta)_h}$ is the ability of resource $(wt_\theta)_h$ to complete activity *i*.

(9) The available time (tr_{wt_θ}) of team wt_θ depends on the available time of the last available resource in the team.

$$tr_{wt_\theta} = Max\left(tr_{(wt_\theta)_h}\right), h = 1,\ldots,|wt_\theta| \tag{9}$$

Where $|wt_\theta|$ refers to the number of resource contained in wt_θ, $(wt_\theta)_h$ refers to resource *h* that is involved in team wt_θ of activity *i*.

(10) The matrix elements of lag matrix *a,* with dimensions *m* × 1, are the set of the presently demanded resource scheduling activities; the values of these elements are expressed as the following:

$$a_i = \begin{cases} 1, \Delta t_i < 0 \\ 0, \Delta t_i > 0 \end{cases} \tag{10}$$

For any activity, when the lag time $\Delta t_i < 0$, yielding a value of element $a_i = 1$ in the lag matrix, the implementation of the activity lags. Otherwise, when $a_i = 0$, activity i is executed normally.

(11) The cost $C_{i(wt\theta)}$ needed for team wt_θ's execution of activity i depends on the execution time of each resource member.

$$C_{i(wt_\theta)} = \sum_{h=1}^{|wt_\theta|} \gamma_{(wt_\theta)_h} \cdot c_{(wt_\theta)_h} \cdot \frac{W_{i(wt_\theta)_h}}{A_{i(wt_\theta)_h}}, h = 1,\ldots,|wt_\theta| \quad (11)$$

Where $\gamma_{(wt_\theta)_h}$ is the modified coefficient of the demanded time for the execution of activity i by resource member h of *team* wt_θ, $w_{i(wt_\theta)_h}$ is the workload executed by resource member h, $A_{i(wt_\theta)_h}$ is the ability of resource member h to complete activity i, and $c_{(wt_\theta)_h}$ is the running cost per unit of the execution of activity i by the resource member.

2.3. Modeling

In a workflow management system, the workflow instance $wf_k(k=1,...,p)$ is running, p is the total number of currently running workflow instances. At a certain moment, the active node $i(i=1,...,m)$ is to be performed and m is the total number of activities to be performed, the resource $j(j=1,...,n)$ is available that can be allocated and n is the total number of resources available. R represents a resource set, $R_i \in R$ represents the resource set that could be allocated to task i, and c_j represents the execution cost of resource j per unit of time.

In order to select the appropriate resource $r_i(r_i=1,...,n)$ for the execution of activity i, thereby ensuring the shortest total activity execution time, the shortest total activity lag time, the fewest lag activities, and the lowest total resource consumption cost, the following formulas were derived:

The shortest total activity execution time is

$$\text{Min } f_1(r_i) = Min \sum_{i=1}^{m} (w_i \cdot (f_{i-1} + t_{i(wt_\theta)})) \quad (12)$$

The shortest total activity lag time is

$$\text{Min } f_2(r_i) = Min \sum_{i=1}^{m} (\alpha_i \cdot w_i \cdot |\Delta t_i|) \quad (13)$$

The fewest lag activities is

$$\text{Min } f_3(r_i) = Min \sum_{i=1}^{m} \alpha_i \tag{14}$$

The lowest total resource consumption cost is

$$\text{Min } f_4(r_i) = Min \sum_{i=1}^{m} C_{i(wt_\theta)} \tag{15}$$

s.t.

$$r_i \neq r_k, if\ i \neq k \tag{16}$$

$$\Delta t_i = te_i - tf_i \tag{17}$$

$$tf_i = ts_i + t_{ij} \tag{18}$$

$$ts_i = tr_{j.k} \tag{19}$$

$$tr_{j.k} = \begin{cases} \sum_{q=0}^{k-1} tp_{jq}, tv_j < tc \\ tc + \sum_{q=0}^{k-1} tp_{jq}, tv_j \geq tc \end{cases} \tag{20}$$

$$t_{ij} = \alpha \cdot \frac{W_i}{A_{ij}} \tag{21}$$

$$t_{i(wt_{\theta})} = Max\left(\gamma_{(wt\theta)h} \cdot \frac{W_{i(wt_{\theta})_h}}{A_{i(wt_{\theta})_h}}\right), h = 1,\ldots,|\, wt_{\theta}\,| \tag{22}$$

$$tr_{wt_{\theta}} = Max\left(tr_{(wt_{\theta})_h}\right), h = 1,\ldots,|\, wt_{\theta}\,| \tag{23}$$

$$a_i = \begin{cases} 1, \Delta t_i < 0 \\ 0, \Delta t_i > 0 \end{cases} \tag{24}$$

where w_i is the importance of the current activity in the running instance w_{fk}.

3. Leapfrog Algorithm

Shufflered leapfrog algorithm (SFLA) belongs to a class of heuristic swarm intelligence optimization algorithms which triggers a heuristic search for an optimal solution using a certain mathematical function (Peng-jun & San-yang, 2009). The algorithm combines the advantages of both Memetic algorithm and particle swarm optimization algorithm; some of the advantages include relatively simple concept, relatively few parameters, strong global optimization capability, high computing speed and robustness (Zhu & Zhang, 2014). Leapfrog algorithm has been successfully applied to some technology fields, such as multi-objective optimization technology, traffic control, cluster analysis, and some combinatorial optimization problems.

In the leapfrog algorithm, the population is composed of many "frogs" with the same structure, each representing a particular solution. The total population is divided into many sub populations called memeplexes. Different sub-populations (memeplexes) can be thought of as collections of "frogs" with different cultures; as such, these memplexes are executed in accordance with certain local search strategies. In each sub-population (memeplex), each "frog" has its own ideas, but is also affected by the others in its community. After a certain memeplex evolution and jumping process, the cultural ideas of the various sub-populations become mixed during computation; the local search and jumping are executed until the convergence criteria are met.

Based on the principle of the leapfrog algorithm, the detailed steps can be stated as the following.

Step 1: Initializing parameters. Selecting the appropriate number *m* of memeplex and frog number *n* of every memeplex, then the number of the population is $F = m \times n$.

Step 2: Initializing population. *F* frogs *X*(1), *X*(2),...,*X*(*F*) are generated randomly in the feasible region Ω, let *d* be a dimension variable, the frog *i* can be represented as X(i) = $\{X_i^1, X_i^2,...,X_i^d\}$ *x*(*i*) and the fitness function is *f*(*i*) = *F*(*X*(i)).

Step 3: *Sequencing frogs. The F frogs are ordered according the fitness values and* storage with, the best frog $X_g = U(1)$ is recorded.

Step 4: Grouping the frogs and storing in different memeplex. Dividing *U* into *m* memeplex, $Y_1, Y_2,\ldots, Y_m$, there are *n* frogs in every memeplex. The frogs are allocated according to the formula

$$Y_j = \left\{ X\left(j\right)_k, f\left(j\right)_k \mid X\left(j\right)_k = X\left(k + m\left(j-1\right)\right), f\left(j\right)_k = f\left(k + m\left(j-1\right)\right), j = 1,\ldots,n \right\}$$

Step 5: Evolving of memeplex. In a memeplex, every forg will be affected by the others and will evolve through the memeplex, the frog will leap toward the targeting positon.

Step 6: Leaping and moving. The frogs will leap and move among the memeplex, after certain Memeplex evolutions are executed in every memeplex, the frog groups $Y_1, Y_2,\ldots, Y_m$ are merged into *U* and reordering is carried out, the best frog P_g of the whole population is updated.

Step 7: Judging the algorithm termination condition. If the iteration meets the termination condition, the iteration end and the best frog is output, otherwise executing step 4 again.

4. Illustrative Example

Figure 2 is a workflow process model of the mold processing of a domestic mold manufacturing enterprise; the definition of each active node is shown in Table 1.

In order to respond quickly to market demands and improve enterprise productivity, the original business process was optimized, and two changes were made to the original workflow process model; the compound workflow model was obtained as shown in Figure 3.

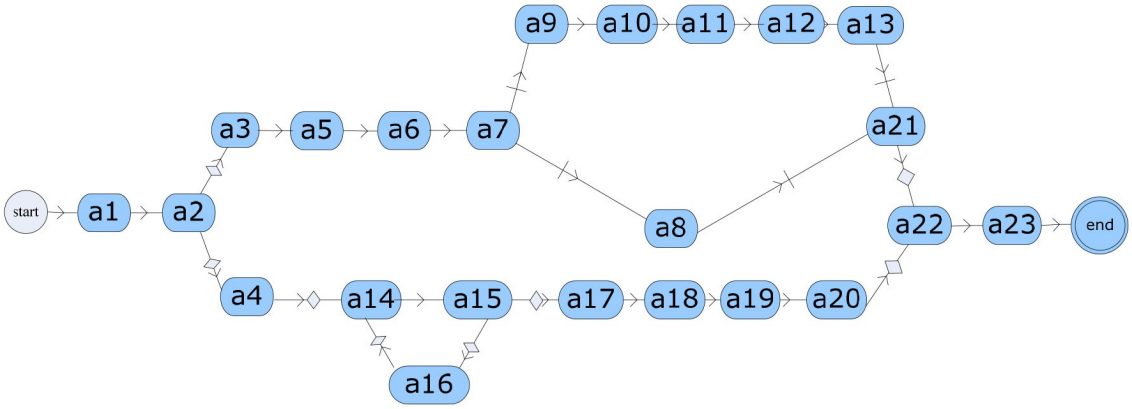

Figure 2. Business process model for mold project management

Node	Node Meaning	Node	Node Meaning
a_1	Conceptual design	a_{13}	Finish-machining the movable / fixed mold
a_2	Structural design	a_{14}	Formulating purchasing plan of mould base
a_3	Detailed design	a_{15}	Manufacturers quoting
a_4	Formulating purchase/customized demands	a_{16}	Adjusting purchase/customized model plan
a_5	Drawing 3d/2d mold figures	a_{17}	Generating purchase /customized orders
a_6	Process design	a_{18}	Signing for purchase / customized model
a_7	Engineering analysis	a_{19}	Financial payments
a_8	Fabricating the self-control parts	a_{20}	Storing the purchased (customized) items
a_9	Rough-machining the movable / fixed mold	a_{21}	Quality inspection
a_{10}	Machining the electrodes	a_{22}	Assembly and bench-work repair
a_{11}	Heat treatment	a_{23}	Mold tryout
a_{12}	Electromachining the movable/fixed mould		

Table 1. Active node definitions

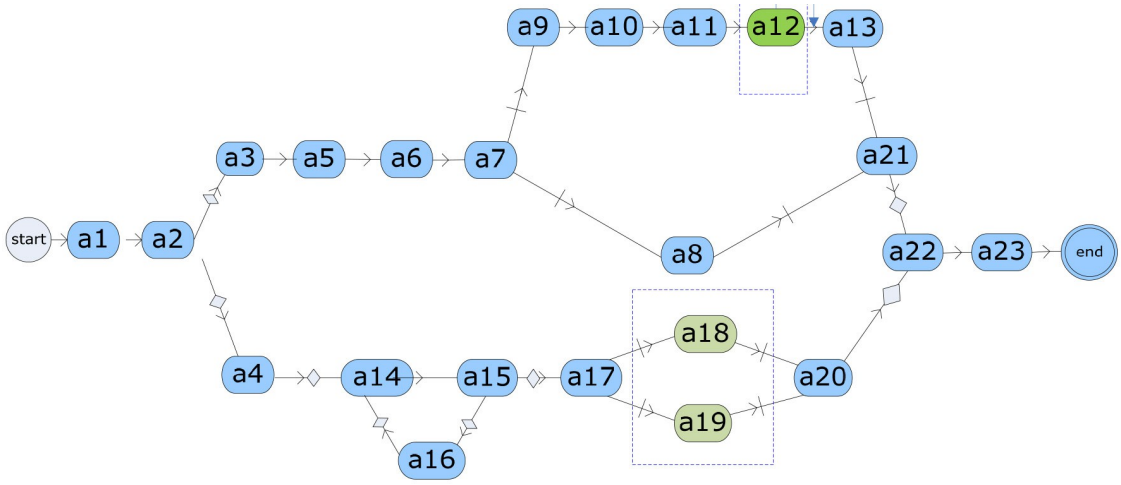

Figure 3. Compound workflow process model

Notes deletion: moving and fixed mold electrical processing (a_{12}), Node structure parallelization: purchase signoff (a_{18}) and purchase order payment (a_{19}) are executed simultaneously.

Suppose a set of migration nodes of the multiple instances is $A_i = \{a_1, a_2, a_3, a_4, a_5, a_6, a_7, a_8, a_9, a_{14}, a_{15}, a_{16}\}$. Table 2 shows the relationships between the allocated active nodes and the time-related resources available for the current scheduling time. In order to facilitate expression and calculation, each of the time parameters in the Table 2 was converted to a relative time scale, and the current scheduling time was set to 0.

The expected time te_i of activity i was used to denote the expected completion time of each activity, and the time resource j spent executing activity i was obtained in accordance with the preference of the resources, the resource capacity and workload of the activity.

Activity i	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{14}	a_{15}	a_{16}	a_{17}
Expected moment te_i Resource j	5	7	9	15	13	21	14	29	26	15	8	5	14
r_1	4	5	8	-	-	-	-	-	-	-	-	-	-
r_2	3	9	11	-	-	-	-	-	-	-	-	-	-
r_3	7	12	15	-	-	-	-	-	-	-	-	-	-
r_4	6	8	12	-	-	-	-	-	-	-	-	-	-
r_5	10	4	10	-	-	-	-	-	-	-	-	-	-
r_6	-	-	-	-	15	22	-	-	-	-	-	-	-
r_7	-	-	-	-	12	24	-	-	-	-	-	-	-
r_8	-	-	-	-	13	-	-	-	-	-	-	-	-
r_9	-	-	-	-	-	-	18	-	-	-	-	-	-
r_{10}	-	-	-	-	-	-	18	-	-	-	-	-	-
r_{11}	-	-	-	10	-	-	-	-	-	-	-	-	13
r_{12}	-	-	-	-	-	-	-	-	-	14	-	5	-
r_{13}	-	-	-	-	-	-	-	-	-	13	-	5	-
r_{14}	-	-	-	-	-	-	-	-	-	-	10	-	-
r_{18}	-	-	-	-	-	-	-	24	25	-	-	-	-
r_{19}	-	-	-	-	-	-	-	30	26	-	-	-	-
r_{20}	-	-	-	-	-	-	-	42	26	-	-	-	-
r_{21}	-	-	-	-	-	-	-	22	-	-	-	-	-

Table 2. Time information for resource-activities

Activity i / Resource j	a_1	a_2	a_3	a_4	a_5	a_6	a_7	a_8	a_9	a_{14}	a_{15}	a_{16}	a_{17}
r_1	0.8	0.9	0.8	-	-	-	-	-	-	-	-	-	-
r_2	0.8	0.9	1.1	-	-	-	-	-	-	-	-	-	-
r_3	0.8	0.9	1.5	-	-	-	-	-	-	-	-	-	-
r_4	0.7	0.8	1.2	-	-	-	-	-	-	-	-	-	-
r_5	0.9	1.0	1.0	-	-	-	-	-	-	-	-	-	-
r_6	-	-	-	-	1.0	0.8	-	-	-	-	-	-	-
r_7	-	-	-	-	1.0	0.9	-	-	-	-	-	-	-
r_8	-	-	-	-	1.0	-	-	-	-	-	-	-	-
r_9	-	-	-	-	-	-	0.8	-	-	-	-	-	-
r_{10}	-	-	-	-	-	-	1.0	-	-	-	-	-	-
r_{11}	-	-	-	0.9	-	-	-	-	-	-	-	-	0.88
r_{12}	-	-	-	-	-	-	-	-	-	1.0	-	0.8	-
r_{13}	-	-	-	-	-	-	-	-	-	1.0	-	0.9	-
r_{14}	-	-	-	-	-	-	-	-	-	-	0.7	-	-
r_{18}	-	-	-	-	-	-	-	1.0	1.1	-	-	-	-
r_{19}	-	-	-	-	-	-	-	1.1	1.0	-	-	-	-
r_{20}	-	-	-	-	-	-	-	1.2	1.1	-	-	-	-
r_{21}	-	-	-	-	-	-	-	1.2	-	-	-	-	-

Table 3. Cost information for resource-activities

The cost that resource *j* spent executing activity *i* per unit of time is shown in Table 3. The cost of resource execution per unit of time was determined by the properties of the resource itself.

In order to develop the optimization mathematical model for scheduling described in section 2.3, the optimal resource scheduling results were solved using the designed leapfrog algorithm. First, the parameters were initialized; the number of frog populations was set to 40, the ethnic memeplex number was set to 5, the number of frogs in each memeplex was set to 8, and the maximum number of iterations was set to 250. Next, the C# programming language was used to write the algorithm program. The process used to solve the scheduling problem is shown in Figure 4, and the fitness value curve of the possible solutions for each generation is shown in Figure 5.

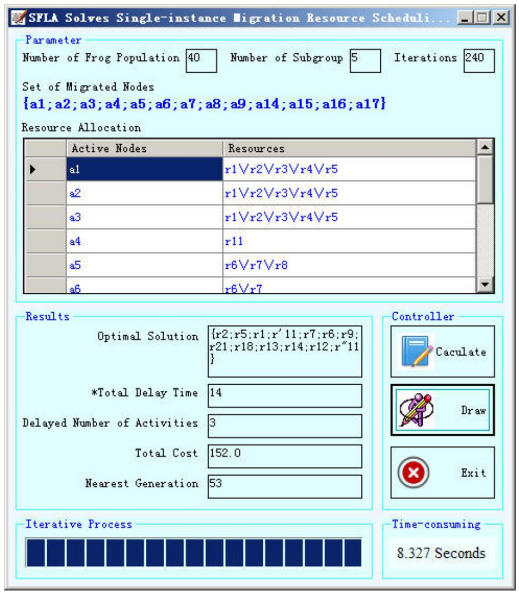

Figure 4. Resource scheduling optimization module for multi-instance migration

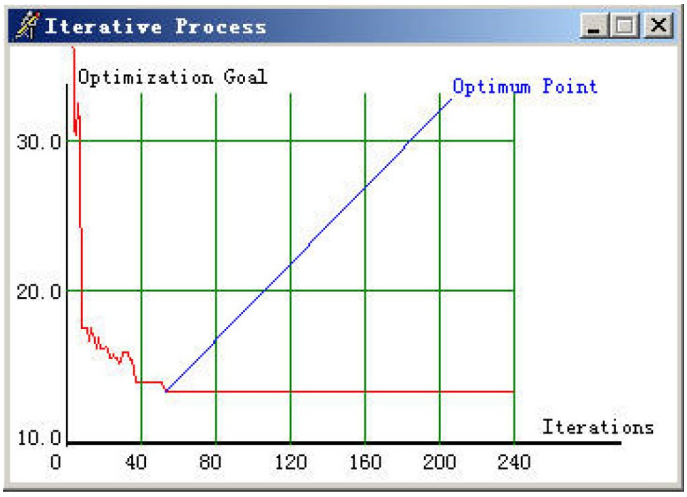

Figure 5. Fitness curve of every solution in each generation

The scheduling result (S) is equal to $\{r_2,r_5,r_1,r^{(1)}_{11},r_7,r_6,r_9,r_{21},r_{18},r_{13},r_{14},r_{12},r^{(2)}_{11}\}$, the total activity lag time is equal to f2({ S3}) = 14, the number of lag activities is equal to f3({S3}) = 3, and the consumption cost is equal to f4({S3}) = 152.0, where $r^{(1)}_{11}$ and $r^{(2)}_{11}$ represent the execution of active nodes a_4 and a_{17} by resource r_{11}, and the order of execution is a_4, a_{17}.

5. Conclusion

In this paper, the resource scheduling optimization of workflow instance migration was analyzed. In order to investigate the resource scheduling problem of workflow multi-instance migration, time and cost relationships of activities and resources were analyzed. Certain factors, including the shortest overall activity execution time, minimum overall activity lag time, minimum number of lag activities, and minimum resource consumption cost were used to establish a workflow resource scheduling optimization mathematical model for workflow instance migration. A leapfrog algorithm was implemented to solve the optimal resource scheduling problem. Specific examples were listed to validate the proposed mathematical resource scheduling model and designed algorithm. The results showed that, under the constraints of resource cost and quantity, an optimal resource scheduling scheme for workflow migration can be found, ensuring a minimal running time and optimal cost. On the other hand, only the same-time working style of multi resources executing workflow activity is considered, case of activity executed with other working style should be further studied. Also, only resource scheduling of multi instance migration in a single workflow model is considered, resource scheduling optimization of multi instance migration in multi workflow models should be further studied.

Acknowledgments

This research is supported by Nature Science Foundation of China (Grant No. 61402361), Technology Project of Shaanxi province, China (Grant No. 14JK1521), Science and Technology Research of Shaanxi province, China (Grant No. 2012KJXX-34), Science and Technology Innovation Team for Youth of Xi'an University of Technology, China (Grant No. 102-211408) and Scientific Research Plan of Xi'an University of Technology, China (Grant No. 102-211305).

References

Deng, T.Q., Ren, G.Q, & Liu, Y.B. (2012). Genetic Algorithm Based Approach to Personal Worklist Resource Scheduling. *Journal of Software*, 23, 1702-1716. http://dx.doi.org/10.3724/SP.J.1001.2012.04222

Feifan, K., & Xuanxi, N. (2006). *Wokflow Modeling Design Based on Petri Net and UML. Journal of Nanjing University of Aeronautics & Astronautics*, 38, 121-125.

Gao, X., Xu, L., Wang, X., Li, Y., Yang, M., & Liu, Y. (2013). Workflow process modelling and resource allocation based on polychromatic sets theory. *Enterprise Information System*, 7, 198–226. http://dx.doi.org/10.1080/17517575.2012.745617

Lijun, Z., Jun, M., & Tao, Y. (2009). Research and implementation of dynamic task scheduling based on workflow. *Computer Engineering and Design*, 30, 2533-2540.

Peng-jun, Z, & San-yang, L. (2009). Shuffled frog leaping algorithm for solving complex functions. *Application Research of Computers*, 26, 2435-2437.

Sakellariou, R., Zbao, H., Tsiakkouri, E., & Dikaiako, M. (2007). Scheduling workflows with budget constraints. *Integrated Research in Grid Computing*, 5, 189-202. http://dx.doi.org/10.1007/978-0-387-47658-2_14

Shengwen, L., & Junfang, G. (2010). *R*esearch on Workflow Schedule Model based on Fuzzy Theory. *Application Research of Computers*, 27, 131-133.

Smanchat, S., Indrawan, M., & Ling, S. (2011). A Scheduler based on Resource Competition for Parameter Sweep Workflow. *Procedia Computer Science*, 4, 176-185. http://dx.doi.org/10.1016/j.procs.2011.04.019

Subo, L., Jianchong, Z., & Haizhu, X. (2011). Resource Scheduling based on Workflow Model. *Fire Control & Command Control*, 36, 126-130.

Tramontina, G.B, & Wainer, J. (2005). Modeling the Behavior of Dispatching Rules in Workflow Systems: A Statistical Approach. *Computer Science*, 8, 208-215.

Xu, L.D. (2014). Advances in e-business engineering management. *Information Technology and Management,* 15, 65-67.

Yang, M., Han, Z., Gao, X., & Liu, Y. (2012). *R*esource Allocation Optimization of Workflow with Multi-instance and Multi-resource Based on Queuing Theory. *International Review on Computers and Software*, 7, 2384-2393.

Yu, J., Buyya, R., & Tham, C.K. (2005). Cost-based Scheduling of Scientific Workflow Applications on Utility Grids. *e-Science and Grid Computing*, 4, 147-154.

Zhu, G.Y., & Zhang, W.B. (2014). An improved Shuffled Frog-leaping Algorithm to optimize component pick-and-place sequencing optimization problem. *Expert Systems with Applications*, 41, 6818-6829. http://dx.doi.org/10.1016/j.eswa.2014.04.038