doi:10.3772/j.issn.1006-6748.2016.03.006

Concurrent processes scheduling with scarce resources in small and medium enterprises[1]

Ma Songhua (马嵩华)[2]*, Tian Ling**
(* School of Mechanical Engineering, Shandong University, Jinan 250061, P. R. China)
(** Department of Mechanical Engineering, Tsinghua University, Beijing 100084, P. R. China)

Abstract

Scarce resources, precedence and non-determined time-lag are three constraints commonly found in small and medium manufacturing enterprises (SMEs), which are deemed to block the application of workflow management system (WfMS). To tackle this problem, a workflow scheduling approach is proposed based on timing workflow net (TWF-net) and genetic algorithm (GA). The workflow is modelled in a form of TWF-net in favour of process simulation and resource conflict checking. After simplifying and reconstructing the set of workflow instance, the conflict resolution problem is transformed into a resource-constrained project scheduling problem (RCPSP), which could be efficiently solved by a heuristic method, such as GA. Finally, problems of various sizes are utilized to test the performance of the proposed algorithm and to compare it with first-come-first-served (FCFS) strategy. The evaluation demonstrates that the proposed method is an overwhelming and effective approach for scheduling the concurrent processes with precedence and resource constraints.

Key words: concurrent processes scheduling, resource constraint, precedence constraint, timing workflow net (TWF-net), genetic algorithm (GA)

0 Introduction

SMEs contribute significantly to a nation's gross domestic product (GDP) and provide employment to a large number of people[1]. However, more and more SMEs are now struggling to survive due to immense pressure created both by globalization and giant multinational companies. Then their failures could lead to large scale unemployment and consequent social tensions. Floyde, et al.[2] have pointed out that the typical problems faced by SMEs include four aspects: a lack of financial resources and managerial skill, weak commitment from managers, a lack of representation by workers and a contingent workforce. Compared with large enterprises, SMEs have a simpler organization dealing with less specialized tasks by scarce staff, financial, and material resources[3]. Thus SMEs may implement Workflow Management System (WfMS) to manage this effectively. A WfMS enables process automation through integrating, coordinating, and communicating between human and automatic activities in processes.

Due to resource limitation, the workflow manager often sets several resource constraints when he/she defines a workflow. In other words, some resources would be shared among different activities. In a WfMS, multiple workflow instances and activities coexist at a period, and the executing processes might be disturbed by the new coming activities in other processes. In this case, the ability of resource assignment plays a critical role in WfMS. It is worth noticing that there is no WfMS specifically designed for scarce resources scheduling. Traditional workflow models focus on process scheduling without considering whether resources are available. Without an effective scheduling method to coordinate limited resources and heavy workload, the current WfMSs are unfavourable to be adopted and implemented by SMEs.

Consequently, the study concerns about two issues in SMEs: one is checking the temporal violation subject to precedence and resource constraints in concurrent processes; the other is scheduling the conflicting activities to minimize the makespan of all workflow in-

[1] Supported by the Postdoctoral Science Foundation of China (No. 2015M572022) and the National Natural Science Foundation of China (No. 51175304).

[2] To whom correspondence should be addressed. E-mail: msh_1216@aliyun.com

Received on May 21, 2015

stances. To reschedule concurrent processes, the TWP-net is applied to model, monitor and reason the workflow with precedence and resource constraints, and further genetic algorithm (GA) is used to resolve resource conflicts to minimize makespan. The procedure is illustrated in Fig. 1.

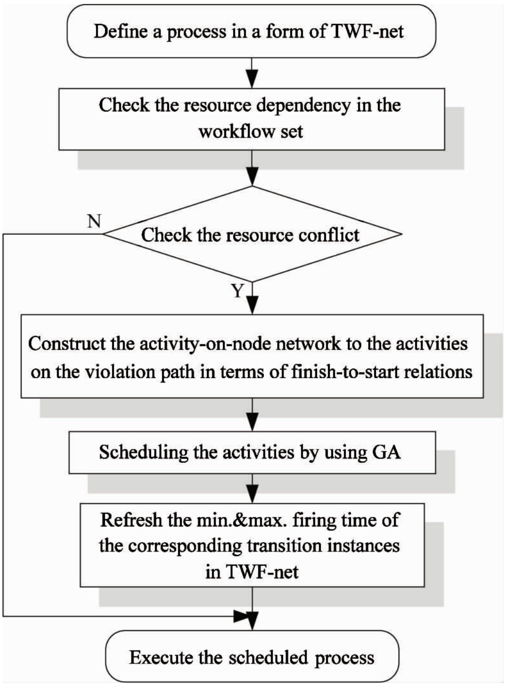

Define a process in a form of TWF-net
Check the resource dependency in the workflow set
Check the resource conflict
N
Y
Construct the activity-on-node network to the activities on the violation path in terms of finish-to-start relations
Scheduling the activities by using GA
Refresh the min.&max. firing time of the corresponding transition instances in TWF-net
Execute the scheduled process

Fig. 1 The proposed approach skeleton

1 Related work

To control workflows in SMEs, the workflow instances should be analyzed to ensure that the activities are consistent about resource at the runtime. The conflicting activities should be rescheduled, then the subsequent activities are postponed, and the process makespan would be extended. Up to now, only a few investigations[4-6] have addressed this problem, and proposed some approaches for the verification of resource consistency in workflows, and further have discussed the solution for the resource conflicts.

1.1 Checking resource conflicts in workflows

Combining timing constraints into workflow models becomes a popular approach to check the resource conflict in a workflow. Zhou and Chen[5], Belhe and Kusiak[7] used Integrated DEFinition for Process Description Capture Method (IDEF3) as the activities network to represent timing constraints. Adam, et al.[8] took timing constraints to analyze structural correctness of a Petri-net based workflow model. Li, et al.[9] made a timing workflow net (TWF-net) for performance analysis. The algorithms of building the timed graph with lower-bound and upper-bound constraints to detect violation of timing constraints were clarified in Ref. [10]. In Ref. [11], the absolute and relative deadline constraints were modelled in a TWF-net, and the dynamic methodologies verifying such constraints were thus built. These approaches could be also used for monitoring workflow execution, reasoning about deadlines, and modifying temporal violation of each activity.

Resource allocation is another topic for analyzing the resource conflict. Park and Reveliotis[12] integrated the resource allocation system into a workflow model in order to analyze deadlock and synchronization problems. Chen, et al.[6] proposed a resource assignment Petri net that contained resource sharing and assigning strategies. Corresponding to a specific sequence of activities, Julia, et al.[13] introduced a hybrid resource (discrete & continuous) allocation mechanism into the p-time Petri net to obtain an acceptable scenario. Wu et al.[14] built a resource allocation procedure that minimizes schedule makespan into sampling scheme for flexible resource flow shop scheduling problem.

In order to support scheduling with resource and precedence constraints in concurrent processes, the relationships between resource allocation and temporal violation need to be clarified, however, none of above work provides such discussion currently.

1.2 Scheduling with resource and precedence constraints

Scheduling is concerned with the optimal allocation of scarce resources to activities over time. The conflicting activities scheduling issues appear as a resource-constrained project scheduling problem (RCPSP). RCPSP consists of activities that must be scheduled subjecting to precedence and resource constraints such that the makespan is minimized. It is known to be NP-hard even when the precedence graph is empty and the resource number equals to 2[15], which has attracted numerous researchers who developed several enumerative and heuristic scheduling procedures.

The enumerative methods such as cutting plane algorithm[16], Lagrangian relaxation technique[17], mixed integer linear model[18] and depth-first search strategy[19], could find optimal schedules inefficiently. The most efficient enumerative method appears to be the branch-and-bound procedure[20,21] which tries to generate a sub-set of the feasible schedules. However they are typically limited due to computing complexity. The heuristic methods are designed to optimize solution in limit time. GA[22,23], ant colony algorithm[24], variable neighbourhood search[25], teaching-learning algorithm[26] and particle swarm optimization[27] are the typical heuristic methods to solve RCPSP.

However, RCPSP is a rather basic model with several assumptions that are too restrictive for many practical applications. Most of the researches performed on machine scheduling did not consider sequence-dependent time-lags to neighbour activities. The time-lags from precursor activity switching to the successor are assumed to zero. Therefore it is hard to find realistic examples about the basic RCPSP. To rich RCPSP, Kim [16, 28] introduced the s-precedence constraints. Bianco and Caramia [20, 29] defined and solved the RCPSP with generalized precedence relations.

2 Workflow model

The terminology of Workflow Management Coalition (WfMC) that defines a process as a network of atomic activities is adopted in our work. The activities are assumed to be executed in a non-preemptive and non-interruptible way. Consequently, while an activity has employed a resource instance, and another activity requests for a resource instance of the same type during runtime, then the latter should either reserve another resource instances which are idle at requested time or undergo awaiting. Here the mentioned resource is a processor such as employee and machine.

2.1 Preliminaries

Definition 1 (PN): A Petri net PN = $<P, T; F>$, where P is a set of places, T is a set of transitions, and $F \subseteq (P \times T) \cup (T \times P)$ is a set of directed arcs linking places and transitions.

A transition is usually represented by a bar, a place represented by a circle, and a token represented by a dot as shown in Fig. 2. Postset (preset) of t is the set of output (input) places of t, denoted by $t\cdot$ and $\cdot t$, respectively. Likewise, postset (preset) of p is the output (input) transition set of p, denoted by $p\cdot$ and $\cdot p$.

Definition 2 (WF-net): A PN is called a workflow net (WF-net) iff the following conditions are true.

(1) PN has two special places: ε and θ. ε is a source, and $\cdot\varepsilon = \varnothing$. θ is a sink, and $\theta\cdot = \varnothing$.

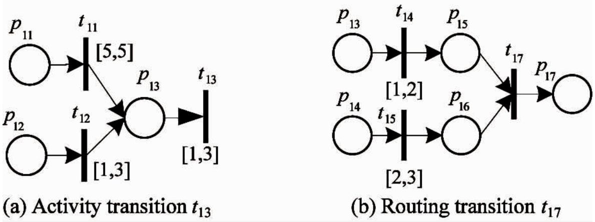
(2) If a new transition is added to PN which connects place θ with ε, i.e., $\cdot t = \{\theta\}$ and $t\cdot = \{\varepsilon\}$, then the resulting PN is strongly connected.

A WF-net gives the process control specification of workflow model. To realize the verification and evaluation in time dimension, its temporal behaviour should be planned. Hence TWF-net is proposed by considering timing constraint with WF-Net.

Definition 3 (TWF-net): A TWF-net is a three tuple $<WF\text{-}net, FI, M>$, where WF-net = $<P, T; F>$; P represents the state of a transaction instance or the condition of its output transitions; T represents activities or routing in workflow; FI denotes a set of delay pairs $[d, D]$ corresponding to each transition, which includes the lower and upper bounds of duration time; $M(p)$ denotes the number of tokens, i.e. transaction instances in p.

Assume that a transition $t \in T$ is enabled in a state $M(\cdot t)$ and associated with a time interval $[d(t), D(t)]$. Then, let $s(t)$ and $\tau(t)$ represent the enabled time and the actual firing time of t, respectively. We have $s(t) + d(t) \leqslant \tau(t) \leqslant s(t) + D(t)$. Generally, TWF-net contains two types of transitions, i.e., activity and routing transitions. The former models the activity in a workflow, while the latter (and-split and and-join) determines the control flow among activities, which is not associated with time interval. In this paper, a workflow instance (process) is an execution case of TWF-net, which begins at the start activity and ends at the end activity. For a workflow specification, each execution case corresponds to a unique TWF-net. Likewise, each execution case of an activity corresponds to a unique transition instance denoted with an identifier in process sets.

Definition 4 (ELFT of activity transition): Let t_i be an activity transition as t_{13} shown in Fig. 2(a), then $FI(t_i)$ represents its time interval. The earliest and the latest firing time (ELFT) of t_i is defined as $\sum(t_i) = \{\sum(t_g) + FI(t_i), \sum(t_{g+1}) + FI(t_i), \cdots, \sum(t_h) + FI(t_i)\}$, where $\cdot(\cdot t_i) = \{t_g, t_{g+1}, \cdots, t_h\}$.



(a) Activity transition t_{13} (b) Routing transition t_{17}

Fig. 2 Transition examples

Definition 5 (ELFT of routing transition): Supposing t_i is a union transition as t_{17} shown in Fig. 2(b), the ELFT of t_i is formulated as $\sum(t_i) = \mathrm{Max}(\sum(t_g), \sum(t_{g+1}), \cdots, \sum(t_h))$, where $\cdot(\cdot t_i) = \{t_g, t_{g+1}, \cdots, t_h\}$.

It is clear that $\sum(t_i)$, where $t_i\cdot = \{\theta\}$, contains the minimum and maximum makespan of the process that t_i belongs to.

2.2 Strategies for checking resource conflicts

If transitions t consumes a resource, then it has a resource constraint denoted as $RC(t)$. In TWF-net graph, the shared resource is denoted by an oval as a resource place p_r, and the corresponding directed arcs to/from the transitions represent the request/release of resource. $M(p_r)$ means the number of resource instances. In this case, an example illustrating resource constraint is shown in Fig. 3, where $RC(t_{12}) = RC(t_{26}) = p_{r1}$.

Definition 6 (**Resource dependency**): Given two activities t_i and $t_j (i \neq j)$, within a workflow or from different workflows, t_i and t_j have a resource dependency if $RC(t_i) \cap RC(t_j) \neq \varnothing$.

If t_i and t_j have a resource dependency, t_i cannot execute simultaneously with t_j at runtime. Otherwise, a resource conflict may arise from competing for the same resource.

Definition 7 (**Resource conflict**): Given a set of concurrent processes, t_i has a resource conflict with t_j if $RC(t_i) \cap RC(t_j) \neq \varnothing$ $((\sum(t_i) \cap \sum(t_j) \neq \varnothing)$ $(M(p_r) \leqslant 1$.

In a WfMS, there are multiple processes running at the same time, where activity t_i and t_j may belong to different processes or even different workflows. Check if the number of required resource instances is less than two, and the difference between the firing time of t_i and the enabled time of t_j is less than zero as shown in Fig. 3. If a temporal resource conflict happens, find the paths with violations in process. In this case, the execution durations of the activities on the paths need to be rescheduled.

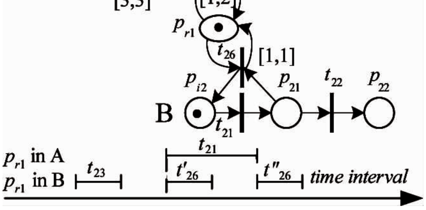


Fig. 3 Resource constraints between activities

2.3 TWF-net simplification

In this paper, the resources have exclusive property, i. e., one resource instance can be used by only one activity without preemption. The first-come-first-served (FCFS) policy could be applied to allocate resource instance to activities. An activity will be postponed for execution until the required resources are available. When the activity finishes, it will release all the occupied resources. Nevertheless, several activities belonging to the same workflow may be enabled at the same time. This case cannot be handled by the FCFS policy. Additionally, the FCFS policy does not make an overall plan to minimize the makespan of all processes. Hence, the other scheduling policies for resolving resource conflicts are significant. Our scheduling policy is proposed in the next section.

To save computing time for the rescheduling of conflicting activities, there are several simplifications to TWF-net. As a preparation to scheduling algorithm, only the activities with resource conflicts on the violation path are kept for scheduling, and their workflow models are simplified. The precedence relations are represented by an activity-on-node network (AN _ net), which is assumed to be acyclic as shown in Fig. 4. The remaining transitions denoted as nodes are renumbered in order. The time-lag is given by summing the time intervals of activities which align between activity j and j' in TWP-net. $l_{jj'}$ is the minimum time-lag in terms of finish-to-start relation between two neighbour activities in the AN _ net. A directed arc from node j to node j' means that activity j precedes j', that is, activity j' cannot start until activity j is finished. l_{0j} is the delayed time of activity j after process starting. In this case, the concurrent processes scheduling problem with precedence and resource constrains is transited into a RCPSP.

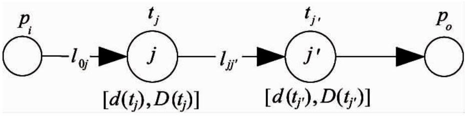


Fig. 4 Representation of a simplified workflow model (AN _ net)

3 Scheduling with precedence and resource constraints

The standard classification scheme for scheduling problems is $\alpha_1 | \alpha_2 | \alpha_3$[30], where α_1 describes the processor structure, α_2 gives the activity characteristics or restrictive requirements, and α_3 defines the objective function to be minimized. By using this scheme, the scheduling problem can be represented in the form:

$$Ph | d_j, l_{jj'}, r_j, prec | \sum T_j, \sum \Delta t_h,$$

where Ph denotes different classes of processors need to be allocated, d_j is the duration of activity j, $l_{jj'}$ denotes the sequence-dependent time-lag between node j and its successor, r_j refers to resource constraints, and $prec$ refers to finish-to-start precedence constraints. The objective function is composed of two sub-functions: makespan and workload balancing. Minimizing the overall makespan of processes is the first optimal objection. Additionally, considering performance, the workload of resources should be balancing during working hours. Then the final objective function is modified into a multi-objective optimization as

$$\min C_{\max} = \min\left(\sum T_j + \sum \Delta t_h\right) \qquad (1)$$

GA has been successfully applied to RCPSP, and easily obtained a near-optimal solution[22,23]. RCPSP is a general scheduling problem. Therefore, GA can be adopted to solve project scheduling, job-shop, flow-shop, open-shop problems and grid computing problem.

For the scheduling problem, the chromosomal representation should be comprised of two parts: activity sequence and processor assignment. Indeed, each chromosome determines the employee by which activity should be processed and the sequence of activities should be processed in advance of the others. It is crucial notice that as chromosomes are being generated, they should be feasible solutions for the problem at hand. Due to precedence constraint, some activities may require not being processed in advance of their precursors. This should be considered when chromosomes are being produced. As the objective function is minimizing $C_{\max}$, to each chromosome i, the fitness value can be defined as

$$fit(i) = 1/C_{\max}(i) \qquad (2)$$

Moreover, each chromosome is assigned a probability value $prob(i)$ which would be used in the selection strategy

$$prob(i) = fit(i)/\sum_i fit(i) \qquad (3)$$

The selection strategy, which influences significantly the performance of GA, is used to select the parents next generation being generated from. Roulette wheel selection (RWS) strategy has been employed by numerous researches and is applied in this study as well.

Algorithm 1 displays the overall procedure of rescheduling the concurrent processes with resource constraints by checking resource conflicts, simplifying workflow models and implementing GA. Given several processes with resource constraints in WfMS, this algorithm begins with TWF _ net reasoning and resource conflicts checking. For each process, all activities on the violation path with resource constraints waiting for assigning are added into the AN _ net. The sequence-dependent time-lags between the remained activities are evaluated according to Definition 4 and 5. Following these preprocessing steps, the concurrent processes scheduling problem is changed into a general RCPSP. GA is used to solve this NP-hard problem. Each chromosome is a compound of activities consequence and processors assignment, which ensures resource and precedence constraints. GA runs until the stopping criterions are met, i. e., a certain number of generations have been performed or the optimal solution have been found. After allocating one activity to one processor, the actual duration of each activity in processes is then updated.

Algorithm 1 Reschedule the activities in workflow instances set

Input: *Procs*, The process set in form of TWF-net with resource constraints

Output: The rescheduled processes with modified $[d(t_i), D(t_i)]$

```
Function reschedule Activities (Procs)
for each eligible activity $t_i$ in Procs
    if $t_i$ is an and-joint routing transition
        $\Sigma(t_i) = \mathrm{Max}(\Sigma(t_g), \Sigma(t_{g+1}), \cdots, \Sigma(t_h))$, where $\cdot(\cdot t_i) = \{t_g, t_{g+1}, \cdots, t_h\}$;
    else if $t_i$ does not have resource constraints with others in Procs
        $\Sigma(t_i) = \{\Sigma(t_g) + FI(t_i), \Sigma(t_{g+1}) + FI(t_i), \cdots, \Sigma(t_h) + FI(t_i)\}$, where $\cdot(\cdot t_i) = \{t_g, t_{g+1}, \cdots, t_h\}$;
    else
        Number $t_i$ and add it into AN _ net;
    end if
end for
for each activity j in AN _ net
    $l_{0j} = \Sigma(t_j)$, $l_{jj'} = l_{0j'} - l_{0j}$;
end for
Initial the chromosome sets;
Calculate the $C_{\max}$ for each chromosome;
while gen < MAXGEN // Maximum generations
    Calculate the fitness value to each chromosome;
    Select the chromosomes by using RWS as the parents to the next generation;
    if random > = XOVR // Crossover rate
        Crossover operation;
    end if
    if random > = MUTR // Mutation rate
        Mutation operation;
    end if
    Calculate the $C_{\max}$ for each chromosome;
    Reinserting the offspring in population;
    Record the optimum value $\min C_{\max}$;
end while
Modify $[d(t_i), D(t_i)]$ according to the chromosome having $\min C_{\max}$;
Return;
```

4 Case study

In this section, a case study is presented to show the verification of the resolution of resource conflicts proposed in this paper. Fig. 5 shows two workflows from a cell phone enterprise in Ref. [11]. It is assumed that there are one and three processes respectively corresponding to the two models. The time-lag of two neighbour activities in AN _ net is defined as the minimum ELFT, which is estimated by using Definition 4 and 5. As preparation, the AN _ net concerning resource constraints for simplified process is shown in Fig. 6. It is worth noting that the activity t_{18} would be implemented twice, and the second execution is denoted as t'_{18} in AN _ net to distinguish them.

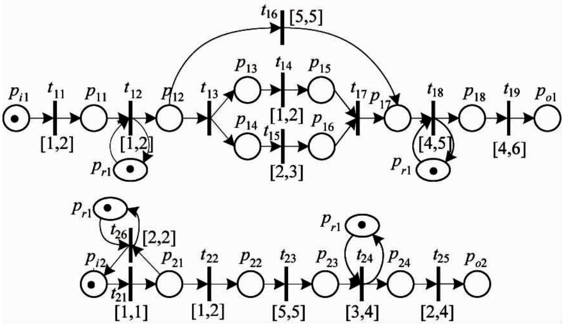


Fig. 5 Two workflow models for the example[11]

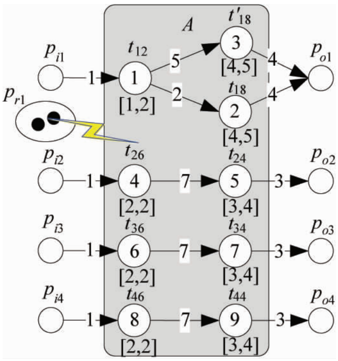
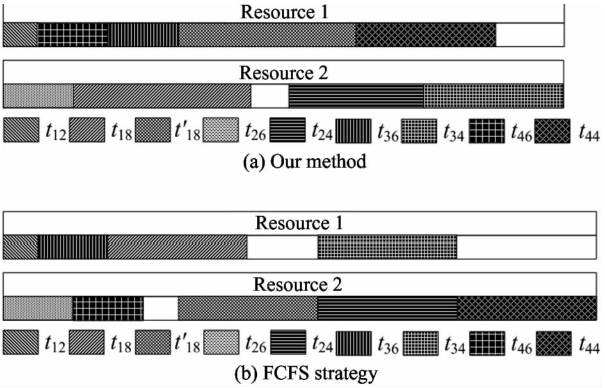


Fig. 6 An AN _ net for four simplified processes

Fig. 7(a) indicates the scheduling results for two processors. In order to show the advantages of the rescheduling strategy for the resource conflicts proposed in this paper, the same case of resource conflicts is resolved by FCFS strategy as shown in Fig. 7(b). Based on the solution matrixes, it can be seen that almost every activity in our method starts earlier than that in FCFS method except activity t_{34}. This indicates that the proposed method can ensure that most of the subsequent activities start as early as possible, although the starting time of some activities is not as early as expected at the beginning of processes. Ensuring that most of the subsequent activities starting as early as possible is one of the advantages of the resolution strategies for resource conflicts proposed in this paper.



(a) Our method

(b) FCFS strategy

Fig. 7 Solution matrix of four processes by two processors

Based on the solution in Table 1, both the overall maximum makespan and delay time solved by our method are shorter than that solved through FCFS strategy. Therefore, ensuring the all processes are finished as early as possible, is the second advantage of our rescheduling strategy for resource conflicts.

Table 1 Comparison of the makespan and delay time of all processes

	$\mid procs \mid = 4, \mid p_r \mid = 2$		$\mid procs \mid = 8, \mid p_r \mid = 3$	
	The proposed method	FCFS strategy	Our method	FCFS strategy
Makespan (Day)	18	19	23	25
Delay time (Day)	3	4	8	10

To the knowledge, the workload balance is a new objective of RCPSP, which is rarely considered in previous researches; nevertheless, it is significant to evaluate the employee performance and to assess the life cycle of machines or tools. The workload of a processor is denoted as the ratio of his working time to system hours. To compare the difference of workload balance between our method and the FCFS strategy, their workload ratio graphs about two processors and four processes are drawn in Fig. 8. Through our method, the workforce is used as efficiently as possible, which is the third advantage proposed in this paper. Based on the aforementioned three advantages, it is indicated that the proposed solution strategy for the resource conflicts among concurrent processes is better than FCFS strategy.

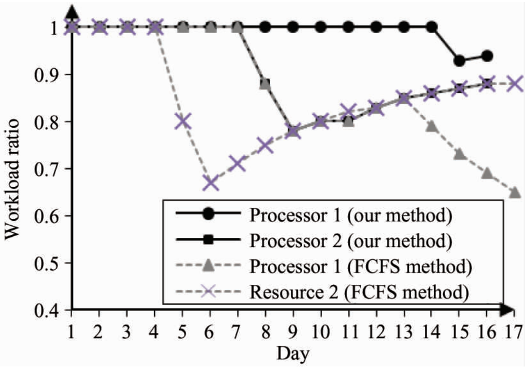


Fig. 8 Comparison of the workload balance of two processors

5 Evaluation

Due to several operations outsourced, the actual workflow models in SMEs are simple as the aforementioned example. In order to evaluate the performance of GA, the RCMPSPLIB library (http://www.eii.uva.es/elena/RCMPSPLIB.htm) is used. As the multi-workflow models to simulate the concurrent processes scheduling problems with several resource constraints, RCMPSPLIB is an open library for the resource constrained multi-project scheduling problem (RCMPSP). The relevant technical details of the implementations are as follows:

(1) As a benchmark, the project scheduling problems with precedence and resource constraints in RCMPSPLIB were selected as workflow models. Two workflow sets (named j3044 _ 3, j6019 _ 6) were taken from the PSPLIB (http://www.om-db.wi.tum.de/psplib/ main.html), with 30 × 6 and 60 × 5 (where 30 × 6 denotes 6 workflow models each one with 30 activities); Three sets with 20 × 3 (named HHH-3, HHH-2, HHH-1) were generated by Browning and Yassine's random generator[31]; Five sets generated by Vázquez et al.[32], i.e., MP1, MP2, MP3, MP4 and MP5, all of which were 10 × 10.

(2) For each workflow model, one process instance was generated. Six types of resources were assigned as processors. The number of each type was respectively set to $M(p_{r1}) = 5$, $M(p_{r2}) = 4$, $M(p_{r3}) = 3$, $M(p_{r4}) = 2$, $M(p_{r5}) = 6$, $M(p_{r6}) = 8$. Each activity had been randomly specified one type of resource during initialization.

(3) GA algorithm was implemented in JAVA, and run in an Intel Core i5 760@2.80GHz processor, with 4GB RAM and executing Windows 7. Each set executed 10 times. The genetic operators were set as: XOVR = 80%, MUTR = 60%, and MAXGEN = 100.

For each execution e of process i, the best value of the objective function $C_{\max}(i, e)$ (which does not include the objective function of workload balance) and the makespan obtained by FCFS strategy $C_{FCFS}(i)$ was stored. For each optimum value obtained by GA, measurements were calculated, i.e.,

$$gap(i, e) = (C_{\max}(i, e) - C_{\mathrm{FCFS}}(i))/C_{\mathrm{FCFS}}(i) \qquad (4)$$

Table 2 presents the average value of $gap(i, e)$ and its standard deviation, summarized by problem class. A graphical analysis of such gaps is found in Fig. 9. The following remarks can be derived:

Table 2 Average and standard deviation of $gap(i, e)$

Workflow set	Activities num. (n)	C_{FCFS}	gap(Average) (%)	gap(SD) (%)
MP1	100	970	−20.1	7.8
MP2	100	1172	−32.5	9.7
MP3	100	852	−10.9	10.3
MP4	100	992	−2.2	8.1
MP5	100	990	−28.8	6.0
HHH-3	60	51	−26.9	11.9
HHH-2	60	40	−0.5	10.5
HHH-1	60	72	−41.4	5.3
j3044 _ 3	180	136	−16.3	6.8
j6019 _ 6	300	203	−19.7	17.5

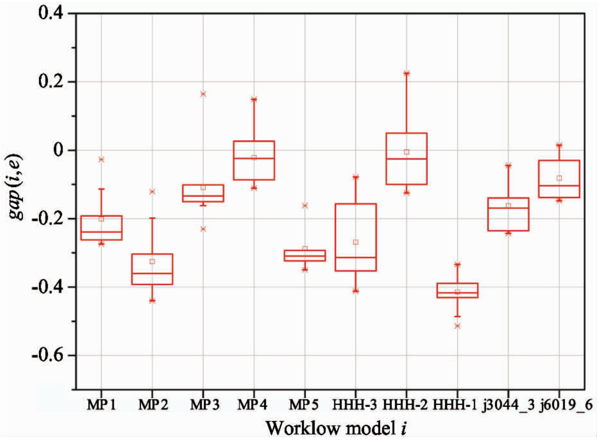


Fig. 9 Graphical analysis of $gap(i, e)$

(1) For the small instances with $n = 60$, seeing Table 2 (average gaps), one can notice that GA reaches better solution than FCFS algorithm. These average gaps are small, and even that of HHH-2 equals to −0.5%. Still regarding these instance sets, Fig. 9 shows that the dispersion of the gaps for $n = 60$ are significantly longer than the others. Consequently, the data presented in Fig. 9 allows one to conclude that the performance of GA is not outstanding compared to that

of FCFS to small instance sets. Conversely, when solving HHH-3 and HHH-1, GA shows better results to most instances.

(2) For medium-sized instances ($n = 100$) and larger instances ($n = 180$ and 300), GA presented better results than FCFS approach, with an acceptable standard deviation.

(3) GA results contain a larger variation of standard deviation in small-size problems. This occurs mainly because of the high proportional impact of small changes on the chromosome, i. e., the relative impact of switch on the positions between two activities in a small-size problem is larger than the same operation on large-size problem.

6 Conclusions

Resource management and verification are important to WfMS. However, scarce employees and mass customization, planning limited resources are especially crucial to raise efficiency in SMEs. In this paper, the main contributions are illustrated as follows:

(1) TWF _ net as workflow model is built constrained by resources, precedence and non-determined time-lag. By reasoning the earliest time to start each activity, a verification approach for resource consistency of workflow is proposed based on TWF _ net.

(2) The rescheduling strategies are based on GA for resource conflicts, which can ensure that most of the subsequent activities start as early as possible and that the overall processes are finished in a shorter time. Additionally, with the objective function of workload balance, there is no overload or leisure to processors.

(3) Since RCPSP is a special RCMPSP, the proposed method can be applied to solve RCMPSP directly without additional modification. The proposed GA is an adaptable scheme in multiprocessor systems.

(4) The proposed method is not a static scheme at building-time, but a dynamic scheduling approach to solve variations in WfMS. In practice, new processes would successively come into WfMS. When AN _ net is updated, the unimplemented activities would be rescheduled. Then the minimum and maximum duration in the specification of finished activities will be replaced by actual implementation time.

How to model human resource and analyze this influence to the resource schedule is our future work. Furthermore a scheduling agent based on our method will be built, which controls resource assignment in WfMS.

Reference

[1] Saini D S, Budhwar P S. Managing the human resource in Indian SMEs: The role of indigenous realities. *Journal of World Business*, 2008, 43(4): 417-434

[2] Floyde A, Lawson G, Shalloe S, et al. The design and implementation of knowledge management systems and e-learning for improved occupational health and safety in small to medium sized enterprises. *Safety Science*, 2013, 60: 69-76

[3] Cragg P, Caldeira M, Ward J. Organizational information systems competences in small and medium-sized enterprises. *Information & Management*, 2011, 48(8): 353-363

[4] Zeng Q, Wang H, Xu D, et al. Conflict detection and resolution for workflows constrained by resources and non-determined durations. *Journal of Systems and Software*, 2008, 81(9): 1491-1504

[5] Zhou Y, Chen Y. Project-oriented resource assignment: from business process modelling to business process instantiation with operational performance consideration. *International Journal of Computer Integrated Manufacturing*, 2008, 21(1): 97-110

[6] Chen Y L, Hsu P Y, Chang Y B. A Petri net approach to support resource assignment in project management. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 2008, 38(3): 564-574

[7] Belhe U, Kusiak A. Resource constrained scheduling of hierarchically structured design activity networks. *IEEE Transactions on Engineering Management*, 1995, 42(2): 150-158

[8] Adam N R, Atluri V, Huang W K. Modeling and analysis of workflows using Petri nets. *Journal of Intelligent Information Systems*, 1998, 10(2): 131-158

[9] Li J, Fan Y, Zhou M. Performance modeling and analysis of workflow. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 2004, 34(2): 229-242

[10] Hsu H J, Wang F J. An incremental analysis for resource conflicts to workflow specifications. *Journal of Systems and Software*, 2008, 81(10): 1770-1783

[11] Du Y, Xiong P, Fan Y, et al. Dynamic checking and solution to temporal violations in concurrent workflow processes. *IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans*, 2011, 41(6): 1166-1181

[12] Park J, Reveliotis S A. Deadlock avoidance in sequential resource allocation systems with multiple resource acquisitions and flexible routings. *IEEE Transactions on Automatic Control*, 2001, 46(10): 1572-1583

[13] Julia S, de Oliveira F F, Valette R. Real time scheduling of workflow management systems based on a p-time Petri net model with hybrid resources. *Simulation Modelling Practice and Theory*, 2008, 16(4): 462-482

[14] Wu W, Wei J, Guan X, et al. A hybrid nested partitions algorithm for scheduling flexible resource in flow shop problem. *International Journal of Production Research*, 2012, 50(10): 2555-2569

[15] Garey M R, Johnson D S. Computers and Intractability: a Guide to the Theory of NP-completeness. New York: W. H. Freeman & Co, 1990. 338

[16] Kim E S. Scheduling of uniform parallel machines with sprecedence constraints. *Mathematical and Computer Modelling*, 2011, 54(1): 576-583

[17] Bianco L, Caramia M. Minimizing the completion time of a project under resource constraints and feeding precedence relations: a Lagrangian relaxation based lower bound. *A Quarterly Journal of Operations Research*, 2011, 9(4): 371-389

[18] Koné O, Artigues C, Lopez P, et al., Event-based MILP models for resource-constrained project scheduling problems. *Computers & Operations Research*, 2011, 38(1): 3-13

[19] Zeballos L. A constraint programming approach to tool allocation and production scheduling in flexible manufacturing systems. *Robotics and Computer-Integrated Manufacturing*, 2010, 26(6): 725-743

[20] Bianco L, Caramia M. An exact algorithm to minimize the makespan in project scheduling with scarce resources and generalized precedence relations. *European Journal of Operational Research*, 2012, 219(1): 73-85

[21] De Reyck B. A branch-and-bound procedure for the resource-constrained project scheduling problem with generalized precedence relations. *European Journal of Operational Research*, 1998, 111(1): 152-174

[22] Çakar T, Köker R, Demir H İ. Parallel robot scheduling to minimize mean tardiness with precedence constraints using a genetic algorithm. *Advances in Engineering Software*, 2008, 39(1): 47-54

[23] Tavakkoli-Moghaddam R, Taheri F, Bazzazi M, et al. Design of a genetic algorithm for bi-objective unrelated parallel machines scheduling with sequence-dependent setup times and precedence constraints. *Computers & Operations Research*, 2009, 36(12): 3224-3230

[24] Lo S T, Chen R M, Huang Y M, et al. Multiprocessor system scheduling with precedence and resource constraints using an enhanced ant colony system. *Expert Systems with Applications*, 2008, 34(3): 2071-2081

[25] Driessel R, Mönch L. Variable neighborhood search approaches for scheduling jobs on parallel machines with sequence-dependent setup times, precedence constraints, and ready times. *Computers & Industrial Engineering*, 2011, 61(2): 336-345

[26] Zheng H, Wang L. An effective teaching-learning-based optimisation algorithm for RCPSP with ordinal interval numbers. *International Journal of Production Research*, 2015, 53(6): 1777-1790

[27] Fahmy A, Hassan T M, Bassioni H. Improving RCPSP solutions quality with stacking justification-application with particle swarm optimization. *Expert Systems with Applications*, 2014, 41(13): 5870-5881

[28] Kim E S, Sung C S, Lee I S. Scheduling of parallel machines to minimize total completion time subject to s-precedence constraints. *Computers & Operations Research*, 2009, 36(3): 698-710

[29] Bianco L, Caramia M. A new lower bound for the resource-constrained project scheduling problem with generalized precedence relations. *Computers & Operations Research*, 2011, 38(1): 14-20

[30] Graham R L, Lawler E L, Lenstra J K, et al., Optimization and approximation in deterministic sequencing and scheduling: a survey. *Annals of Discrete Mathematics*. v5, 1977: 287-326

[31] Browning T R, Yassine A A. Resource-constrained multi-project scheduling: Priority rule performance revisited. *International Journal of Production Economics*, 2010, 126(2): 212-228

[32] Vázquez E P, Calvo M P, Ordóñez P M. Learning process on priority rules to solve the RCMPSP. *Journal of Intelligent Manufacturing*, 2015, 26(1): 123-138

Ma Songhua, born in 1985, is an assistant professor at Shandong University. She received her Ph. D degree from Tsinghua University in 2014. She also received her B. S. degree from Shandong University in 2008. Her research focuses on key technologies for computer integrated manufacturing.