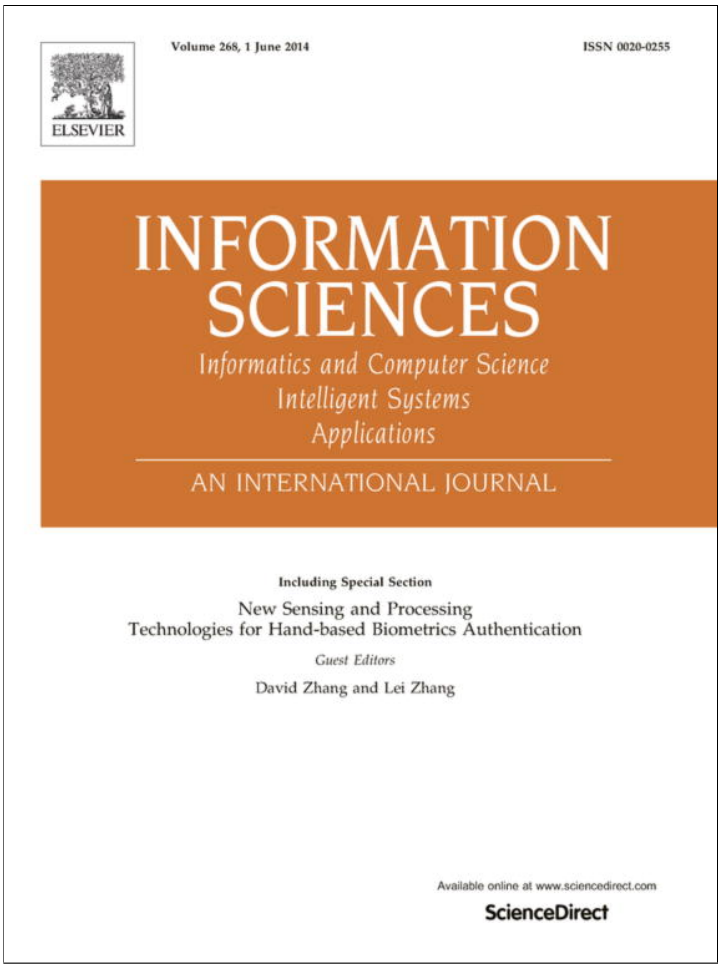

Volume 268, 1 June 2014
ISSN 0020-0255
ELSEVIER
INFORMATION SCIENCES
Informatics and Computer Science
Intelligent Systems
Applications
AN INTERNATIONAL JOURNAL
Including Special Section
New Sensing and Processing Technologies for Hand-based Biometrics Authentication
Guest Editors
David Zhang and Lei Zhang
Available online at www.sciencedirect.com
ScienceDirect

Information Sciences 268 (2014) 357–369

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Planning of business process execution in Business Process Management environments

Hyerim Bae [a], Sanghyup Lee [b], Ilkyeong Moon [c,*]

[a] Department of Industrial Engineering, Pusan National University, Geumjeong-gu, Busan 609-735, Republic of Korea
[b] Automation Research Department, Industrial Research Institute, Hyundai Heavy Industries Co. LTD., Ulsan 682-792, Republic of Korea
[c] Department of Industrial Engineering, Seoul National University, Gwanak-gu, Seoul 151-744, Republic of Korea

A R T I C L E   I N F O

Article history:
Received 2 March 2013
Received in revised form 4 November 2013
Accepted 30 December 2013
Available online 11 January 2014

Keywords:
Business Process Management
Scheduling
Mixed integer programming
Genetic algorithm
Process efficiency

A B S T R A C T

Efficient management of business processes is a key element of enterprise information systems for organizations operating in a competitive business environment. Despite methodology introduced to enhance the effectiveness of Business Process Management, research on the initial phase of system implementation has typically focused on the accurate execution of processes, not efficiency. The enhancement of process efficiency in various manufacturing applications, however, has received much attention over the past several decades. Unfortunately, due to the dissimilarities between business and manufacturing processes, optimized manufacturing processes cannot be applied directly to business processes. This study introduces a methodology for incorporating business process semantics and alternative paths in the Business Process Management structure. The approach entails mixed integer programming (MIP) formulation for a business-process execution plan and a meta-heuristic algorithm to obtain good solutions for multi-activity processes.

1. Introduction

The increasingly competitive business environment is impelling firms to pursue various innovative means to reduce costs and satisfy customer demands. Recently, companies have recognized the importance of Business Process (BP) innovation to survive and accordingly have adopted various methodologies entailing primarily process-oriented reconfigurations of enterprise information systems [4,7,8,15,31,36]. At the core of the new structure, the Business Process Management System (BPMS), comprised of integrated software, designs BP models, manages the execution of them, and facilitates process improvement [11,18]. Thus far, the BPMS has been applied to a variety of areas such as precise modeling, control, and execution of complex business processes [1,23,26]. Moreover, as more companies adopt the BPMS and as the number of processes managed by the system increases, the efficiency aspect of BP execution is garnering more attention. Despite the importance of BP efficiency, optimization methods have rarely been studied, and only scheduling techniques have been extensively investigated.

For over several decades, process efficiency enhancement has been a topic of interest in various manufacturing applications [6,21,27,29,37,43]. These methods, however, have not been applied to BPs, which are different than those used in manufacturing. Most of the differences reflect the human element inherent in BPs, which generates uncontrollable variances and uncertainties as knowledge must be considered in performance evaluations. Applications of the manufacturing methodology

* Corresponding author. Tel.: +82 2 880 7151; fax: +82 2 889 8560.
E-mail address: ikmoon@snu.ac.kr (I. Moon).

0020-0255/$ - see front matter © 2014 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.ins.2013.12.061

to service systems consisting of business process have been recently reported [2,39]. This indirectly shows the necessity of applying the scheduling methodology to BP planning. This paper provides a mathematical formulation and solutions to the BP execution plan by using scheduling algorithms developed for manufacturing systems.

By applying BPMS, companies can solve BP problems with the aid of a mathematical formulation. First, a formal BP model is developed with a specific design tool, and then potential performers for each BP task are determined, much like resource assignments are made in a manufacturing process. Second, due to the automated process flow using a BPMS, the delay between tasks can be reduced significantly during BP execution. This introduced efficiency reduces uncontrollable variance in processing time due to poor task performance by human workers. Third, a process modeling tool in the BPMS provides an interface requesting a user to input expected processing times for each BP activity. By utilizing a BPMS, firms treat the BP like a complicated manufacturing process.

The problem of executing a BPMS is similar to those involved in scheduling manufacturing processes: The objective is to find an optimal assignment that yields the most efficient performance under resource limitations. Efficiency measures commonly used in manufacturing scheduling, such as makespan, cycle time, and flow time, can be directly utilized for enhancing BP efficiency. However, optimization techniques of manufacturing cannot be directly applied to BPs, which allow for alternative paths. Despite the difference between the manufacturing and business environments, by modifying existing scheduling optimization techniques, a good plan of BP execution can be obtained.

This paper is organized as follows. In Section 2, the BP and related execution plan are outlined and compared with a manufacturing process, and existing literature is reviewed. Section 3 offers a definition of a formal BP model along with a mathematical formulation for optimizing a BP execution plan. Section 4 introduces a Genetic Algorithm (GA) effective for finding a solution to the problem. Section 5 discusses the experimentation conducted, and Section 6 draws conclusions.

2. Background and literature review

2.1. Business process execution plan

In an environment where a BPMS is employed, two phases characterize the BP process; build-time for modeling and run-time for execution. During the build-time phase, activities in a process are defined and precedence relationships among the activities are designed with a graph notation [40]. Many BP modeling standards have been suggested to support the build-time phase [24]. In process models, attributes for each activity, such as name, expected execution time, and candidate performer (s) are also specified. During the run-time phase, the values of the attributes are determined as results of process execution. For instance, even though the expected execution time is predicted, actual execution time of the activity is fixed during the run-time phase. Completion times of all activities and the whole process are influenced by the assignment of performers to tasks. In this paper, we call this assignment "the BP execution plan".

Comparing a BP execution problem with a production scheduling problem, one expects the activity execution time to provide a solution for minimizing the completion time of a process. Build-time modules of most commercial BPMSs provide a Graphic User Interface (GUI) in which users input the expected execution time of activities. Using a GUI for the BPMS process, activity, and task performer (i.e., human or software agent), firms can map the BP to the job concept, operation, and machine (of the manufacturing domain), respectively, as shown in Table 1. This mapping has already been introduced in [5], and we added the mapping of execution time to processing time in the table. Even though BP factors are utilized similarly to those characterizing a manufacturing problem, the alternative activity performers and possible paths in a BP execution plan must be considered. Recent models have allowed for alternative machines in the manufacturing problem, which legitimizes the use of scheduling techniques in BP environments. Consideration of alternative paths in process network structures using a mathematical formulation, however, is a unique contribution of this paper; this approach has not been tried in previously published research.

If BPs are automated and managed by a BPMS, the problems are similar to those of manufacturing processes. Therefore, the following reasons explain that a BP execution plan can be developed using the scheduling approach for manufacturing processes:

- The BPMS enforces the use of a formal process model. In a conventional scheduling problem, an operation sequence is given. Where a BPMS is employed, the structure provided in the model clearly depicts the order of precedence among process activities. For each activity, a user (or other expert) can provide an expected processing time, which is used as an input value for scheduling.

Table 1
Mapping between BP execution plans and manufacturing scheduling.

BP execution plan	Manufacturing scheduling
Process (instance)	Job
Activity	Operation
Human/software agent	Machine
Execution time	Processing time

- The BPMS specifies the resources available for each activity. When designing a BPMS model using a process design tool, the individual specifies the group of users who can execute the activity. Furthermore, at run time, the activity can be exclusively assigned to a single user with the sole authority to execute it.
- The BPMS removes unnecessary delays and reduces variances. As soon as an activity is completed, the BPMS delivers the next activity to the subsequent person and provides various functions for facilitating process execution, such as a notification and an alert.

Although BPMSs make the BP execution environment similar to that of manufacturing, BPs still differ from their manufacturing counterparts in that they allow for multiple ways of completing a single objective. Therefore, to develop a process execution plan for a BP, alternative paths should be considered. In this paper, we outline a method for preparing an efficient BP execution plan that acknowledges alternative paths.

2.2. Literature review

As the importance of BP increases, much research has been aimed at improving BP efficiency. Rhee et al. [32] argued that BP efficiency can be influenced by two different means: the Process Engine Perspective (PEP) and the Task Performer Perspective (TPP). From the PEP, they developed a method based on a Theory of Constraints (TOC) of process execution that contributes to efficiency enhancement by controlling task allocation to an overloaded participant and synchronizing the release of the process instance with that individual's pace of work [34]. From the TPP, Rhee et al. [33] calculated the slack times of tasks in workflow processes to improve BP efficiency. The slack time guides task performers such that they tackle urgent tasks first and thus eventually improve BP efficiency. Huang et al. [20] proposed resource behavior measures based on four important aspects for improving BP execution: preference, availability, competence, and cooperation. They showed that this four-part measure can be effectively applicable in a health-care case.

Other approaches to BP efficiency, such as task distribution, priority setting, and evaluation models have been created as well. Ha et al. [14] developed a process execution rule that balances the workload of agents, each of whom has a work list. Kumar et al. [25] dealt with the trade-off problem between (a) observing a deadline and offering work items to an overloaded participant at run time and (b) generating a systematic approach that creates a dynamic balance between quality and performance in workflow systems. Regarding priority settings, Eder et al. [12] employed a personnel schedule that provides information on future work. They showed that with this information, companies can reduce both the turnaround time and rate of time-constraint violations. To achieve the goal of improving BP performance, Han et al. [16] adopted a multi-stage approach. They introduced a two-stage analysis method combining a process-based performance measurement framework (PPMF) and BP simulation. Cho and Lee [10] conducted a study on process-evaluation criteria based on a balanced scorecard approach that used a fuzzy Analytic Hierarchy Process (AHP) method. Oh et al. [30] developed a heuristic algorithm for generation of an execution plan for collaborative business processes. Hofacker and Vetchera [17] represented the business process design as a precedence network of activities and resources allocation. In order to optimize various objective functions for the process design problem, they provided a mathematical programming and compared the results with those of a branch and bound method and GA.

Several studies show scheduling methods to be the most practical approaches for improving workflow efficiency. Baggio et al. [5] provided a method that applies scheduling techniques to minimize the number of late jobs in a workflow. Chang et al. [9] proposed the identification and analysis of critical paths for the management of time and resources within a workflow process. Zhao and Stohr [44] introduced a means of predicting the turnaround time of a time-driven process and allocating the expected processing time of each activity in the process. Kumar and Zhao [26] devised a general framework for efficient workflow management. Senkul and Toroslu [35] proposed architecture for workflow scheduling under resource allocation constraints, not to find an optimal solution but to find a feasible one that reduces the complexity of the workflow scheduling problem by means of a specification language and a scheduler module. Julia et al. [22] introduced an approach based on a *p*-time Petri net model with hybrid resources to solve the real-time scheduling problem in a workflow domain. Yuan et al. [42] developed an efficient heuristic for the cost optimization for workflow scheduling. Want et al. [38] proposed a concept of reliability-driven reputation in a workflow application, and provided a look-ahead genetic algorithm to optimize makespan and reliability of workflow for the problem. Xiao and Ming [41] provided an algorithm for the workflow scheduling problem based on colored petri-net. Recently, some studies on the scheduling problems of workflow in the areas such as grid computing and scientific workflow have been introduced [13,19,28]. Though these approaches are related to this paper in that they studied the workflow scheduling, they introduced only conceptual framework or the methods proposed are too specific to be extended to general BP scheduling problems.

Although all of these prior research efforts provide a good basis for the present study, the need to develop a method that can contribute to BP efficiency remained. Because none of the previous research has provided a mathematical formulation for optimizing BP execution plans, their solutions cannot be evaluated in relation to optimal solutions. In this research, we provide a mathematical formulation for a BP execution plan to optimize various efficiency measures incorporating process structures that include alternative paths.

3. Mixed integer program

3.1. Process model

To develop an optimal BP execution plan, a formal process model is required. In the present study, we employ a process model introduced by Bae et al. [3], and we use their notation in our mathematical formulations.

Definition 1. We define a set of processes, $P = \{p_i | i = 1, 2, 3, \ldots, I\}$, where p_i is a process and I is the number of processes in the system. A process p_i is defined as a tuple of $\langle A_i, L_i \rangle$ and the labeling functions f_s and f_m, each of which is defined as follows:

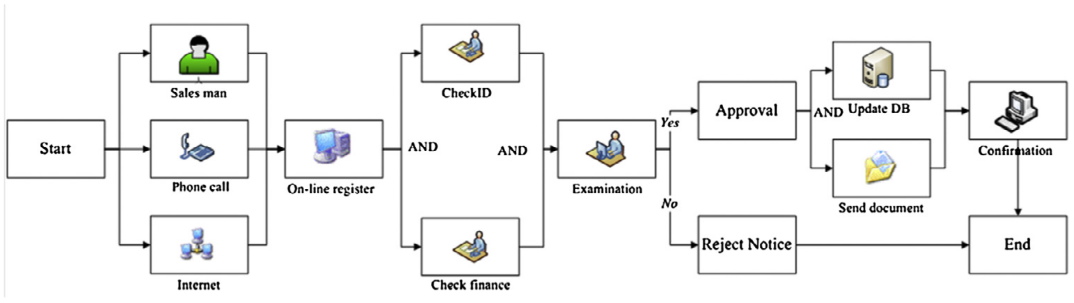
- $A_i = \{a_{ij} | j = 1 \ldots J_i\}$ is a set of activities, where a_{ij} is the jth activity of p_i and J_i is the total number of activities in p_i.
- $L_i = \{(a_{ij^-}, a_{ij^+}) | a_{ij^-}, a_{ij^+} \in A\}$ is a set of links, where (a_{ij^-}, a_{ij^+}) implies that a_{ij^-} immediately precedes a_{ij^+}.
- For a split activity a_{ij} such that $|S_{ij}| > 1$, where $S_{ij} = \{a_{ij^+} | (a_{ij}, a_{ij^+}) \in L_i\}$, $f_s(a_{ij}) =$ ‘AND’ if all a_{ij^+} s should be executed; otherwise, $f_s(a_{ij})=$ ‘OR’.
- For a merge activity a_{ij} such that $|P_{ij}| > 1$, where $P_{ij} = \{a_{ij^-} | (a_{ij^-}, a_{ij}) \in L_i\}$, $f_m(a_{ij}) =$ ‘AND’ if all a_{ij^-} s should be executed; otherwise, $f_m(a_{ij})=$ ‘OR’.

The process model above provides a notation that is required both for representing process structures and for Mixed Integer Program (MIP) formulation in the next section. To develop a mathematical formulation, additional notation is used in detecting split and merge points in ‘AND’ or ‘OR’ blocks. We employed the branch-water procedure [3] and computed the water levels ($w(a_{ij})$) for the activities used to ascertain the split and merge activities. Fig. 1(a) is an example of a business process, and (b) illustrates the business process model for it using the notation of Definition 1.

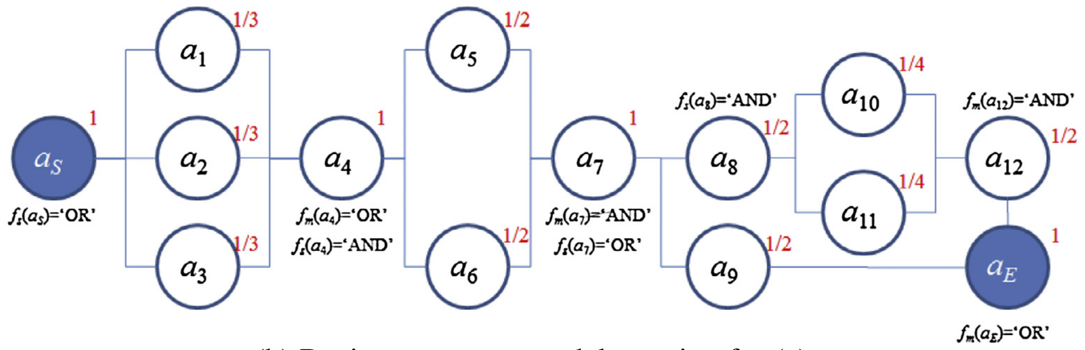
In the example process, it shows a procedure of an insurance application. Insurance application can be initiated by one of a sales person, phone call, internet, and the application method is registered. Then both ID and finance of the applicant are checked and evaluated. After the evaluation, the application is either approved or rejected. In the case of approval, database is updated and documents are sent. Finally all the procedure completes after confirmation. The example process can be converted into a process model shown in Fig. 1 (b), where the activity number, semantic of split and merge, and the value of water level are specified.

3.2. MIP formulation

The following notation is used for formulating the problem mathematically using notation defined in Definition 1:



(a) A business process example: insurance application



(b) Business process model notation for (a)

Fig. 1. A business process example and the notation.

i	index for processes (i = 1, 2, . . ., I)
j	index for activities (j = 1, 2, . . ., J_i)
J_i	number of activities in the ith process
k	index for users or soft agents as activity performers (k = 1, 2, . . ., K)
p	index for the order of processing activities for the kth activity performer (p = 1, 2, . . ., N_k)
$A(k)$	set of activities that are assigned to the kth performer
N_k	number of activities that belong to the set $A(k)$
s_{ij}	starting time of activity a_{ij}
F_{ij}	finishing time of activity a_{ij}
R_{kp}	release time of the kth performer after processing the pth activity in $A(k)$
t_{ijk}	processing time of activity a_{ij} by the kth resource (activity performer)
r_i	ready time of the ith process
H	a large positive number

$$\text{Min} \sum_{i=1}^{I} s_{ij_i} + \sum_{\{k:a_{ij_i}\in A(k)\}} \sum_{p=1}^{N_k} t_{ij_ik} X_{ij_ikp}$$

$$\text{Subject to } s_{ij^-} + \sum_{\{k:a_{ij^-}\in A(k)\}} \sum_{p=1}^{N_k} t_{ij^-k} X_{ij^-kp} \leqslant s_{ij^+},\quad \forall_i, \forall(a_{ij^-}, a_{ij^+}),\ \ j^-, j^+ = 1,2,\ldots J_i \tag{1}$$

$$r_i \leqslant s_{ij},\ \ \forall_i,\ (a_{ij^-}, a_{ij}) \notin L,\ \forall_j,\ j^- = 1,2,\ldots J_i \tag{2}$$

$$s_{ij} + (t_{ijk} + H)X_{ijkp} - R_{kp} \leqslant H,\ \ \{(i,j): a_{ij} \in A(k)\},\ p = 1,2,\ldots,N_k - 1,\ \forall_k \tag{3}$$

$$R_{kp} + HX_{ijkp+1} - s_{ij} \leqslant H, \{(i,j): a_{ij} \in A(k)\},\ \ p = 1,2,\ldots,N_k - 1,\ \forall_k \tag{4}$$

$$R_{kp} - R_{kp+1} \leqslant 0,\ \ p = 1,2,\ldots,N_k - 1,\ \forall_k \tag{5}$$

$$\sum_{\{(i,j):a_{ij}\in A(k)\}} X_{ijkp} - \sum_{\{(i,j):a_{ij}\in A(k)\}} X_{ijkp+1} \geqslant 0,\ \ p = 1,2,\ldots,N_k - 1,\ \forall_k \tag{6}$$

$$\sum_{\{(i,j):a_{ij}\in A(k)\}} X_{ijkp} \leqslant 1,\ \ \forall_k,\ p = 1,2,\ldots,N_k \tag{7}$$

$$\sum_{\{k:a_{ij}\in A(k)\}} \sum_{p=1}^{N_k} X_{ijkp} = 1,\ \ \forall_i,\ \forall_{j^-}, (a_{ij^-}, a_{ij}) \notin L,\ \forall_j,\ j^- = 1,2,\ldots J_i \tag{8}$$

$$\sum_{\{k:a_{ij^-}\in A(k)\}} \sum_{p=1}^{N_k} X_{ij^-kp} - \sum_{\{k:a_{ij^+}\in A(k)\}} \sum_{p=1}^{N_k} X_{ij^+kp} = 0,\ \ \forall_i,\ \forall(a_{ij^-}, a_{ij^+}) \in L,\ f(a_{ij^-}) \neq {}'OR',\ |P_{ij^+}| = 1,\ j^-, j^+ = 1,\ldots J_i \tag{9}$$

$$\sum_{\{j^+|(a_{ij^-}, a_{ij^+})\in L\}} \sum_{\{k:a_{ij^+}\in A(k)\}} \sum_{p=1}^{N_k} X_{ij^+kp} - \sum_{\{k:a_{ij^-}\in A(k)\}} \sum_{p=1}^{N_k} X_{ij^-kp} = 0,\ \ \forall_i,\ \forall_{j^-} f(a_{ij^-}) = {}'OR',\ |S_{ij^-}| > 1 \tag{10}$$

$$\sum_{\{k:a_{ij^-}\in A(k)\}} \sum_{p=1}^{N_k} X_{ij^-kp} - \sum_{\{k:a_{ij^+}\in A(k)\}} \sum_{p=1}^{N_k} X_{ij^+kp} = 0, \forall_i, |S_{ij^-}| > 1, |P_{ij^+}| > 1 f_s(a_{ij^-}) = f_m(a_{ij^+}) \text{ and } w(a_{ij^-}) = w(a_{ij^+}) \tag{11}$$

$$\sum_{\{j^-|(a_{ij^-}, a_{ij^+})\in L\}} \sum_{\{k:o_{ij^-}\in A(k)\}} \sum_{p=1}^{N_k} X_{ij^-kp} - \sum_{\{k:o_{ij^{m+}}\in G(k)\}} \sum_{p=1}^{N_k} X_{ij^{m+}kp} = 0,\ \ \forall_i,\ \forall(a_{ij^-}, a_{ij^+}) \in L,\ \ \text{such that},\ |P_{ij^+}| > 1,\ f_m(a_{ij^+}) = {}'OR' \tag{12}$$

$$X_{ijkp} = 0\ \ or\ 1,\ \ \text{for all } i,j,k,p \tag{13}$$

For the formulation above, the objective is to minimize the mean flow time. The constraints are expressed by Eqs. (1)–(13). The objective in the formulation is to minimize the flow time of each process. Constraint (1) puts forth the requirement that if two activities (a_{ij^-} and a_{ij^+}) have immediate precedence relations in a process, a_{ij^+} can begin only after a_{ij^-} has been completed. The starting time of the first activity of a process must be later than or equal to the ready time of the process, which is represented in Constraint (2). The relationship between the release time of a resource and the completion time of an activity that is assigned to the resource is reflected in Constraint (3). Note that the constraint becomes inactive if X_{ijkp} = 0. Constraint (4) states that the starting time of an activity, which has been assigned to resource k in the (p + 1)th order, must be greater than or equal to the release time of the resource after the pth activity is finished. The release time of resource k after its (p + 1)th activity is finished must be greater than or equal to that of the pth activity, which is reflected in Constraint (5). Constraint (6) requires that the (p + 1)th activity that belongs to $A(k)$ can be assigned only after the pth activity has been assigned to resource$k$. The assumption that only one activity can be assigned to a resource simultaneously is reflected in Constraint (7). Constraint (8) imposes the rule that the first activity in a process must be undertaken. According to Constraint (9), if two activities (a_{ij^-} and a_{ij^+}) have immediate precedence relations and the function of activity a_{ij^+} is not a merge activity in a process, a_{ij^+} can be assigned one of the alternative resources only after an immediately preceding activity has been assigned to a resource. If a process is split at an activity with the 'OR' semantic, then one of its immediately subsequent activities can be conducted. This assumption is reflected in Constraint (10). Regardless of its function, a merge activity a_{ij^+} can be assigned after its corresponding split activity has been assigned to a resource, which is reflected in Constraint (11). If a_{ij^+} is an 'OR' merge activity, it can be assigned to a resource after one of its immediately preceding activities has been conducted; this requirement is reflected in Constraint (12). Constraint (13) directs that an activity can be assigned only to one of its alternative resources.

In this MIP formulation, the user can choose to change the objective function. For example, to minimize the makespan, the user can modify the previous formulation as follows:

$$\text{Min } Z$$

$$s_{iJ_i} + \sum_{\{k:a_{iJ_i}\in A(k)\}} \sum_{p=1}^{N_k} t_{iJ_ik} X_{iJ_ikp} = F_{iJ_i}, \quad \text{for all } i$$

$$F_{iJi} \leqslant Z, \quad \text{for all } i$$

$$\text{Subject to } (1)-(13)$$

In the constraints above, F_{iJi} is the finishing time of the last activity (a_{Ji}) of ith process. A user who wants to minimize the maximum lateness among the individual processes, in relation to their specified due dates, can modify the MIP formulation as follows:

$$\text{Min } \mathrm{L}_{\max}$$

$$S_{iJ_I} + \sum_{\{k:a_{iJi}\in A(k)\}} \sum_{p=1}^{N_k} t_{iJ_ik} X_{iJ_Ikp} - L_i = D_i, \quad \text{for all } i$$

$$L_i \leqslant L_{\max}, \quad \text{for all } i$$

$$\text{Subject to } (1)-(13)$$

4. Genetic algorithm

4.1. Chromosome design and decoding

In this section, we present the design of a chromosome for use in an evolutionary algorithm such as the GA and tabu search techniques. We also delineate the procedure for decoding the designed chromosome. In our evolutionary algorithm, we search for each activity's assignment to one of its alternative resources and also for each resource's order of execution. Therefore, the chromosome is composed of two parts as shown in Fig. 2. The first part shows the assignment of alternative resources, and the second part shows the relative order of processes among activities that use the same resource.

The length of each chromosome equals the total number of activities. The gene value of a sequence chromosome is either a positive or a negative integer. The gene value of an assignment chromosome refers to one of the alternative resources for each activity.

The Chromosome Decoding Procedure (CDP) for a prescribed sequence and assignment chromosome is as follows:

$\overline{v_{ij}}$	the sequence priority chromosome value for activity a_{ij}
$\underline{v_{ij}}$	the resource assignment chromosome value for activity a_{ij}
$\overline{S}$	the set of all unscheduled activities
S	the set of available activities
$P(a_{ij})$	the set of preceding activities for a_{ij} that are not scheduled
Rr_k	the release time of resource k (time at which resource k becomes available)
As_{ij}	possible starting time of a_{ij}

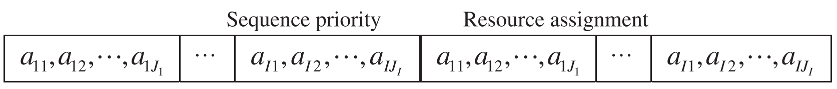
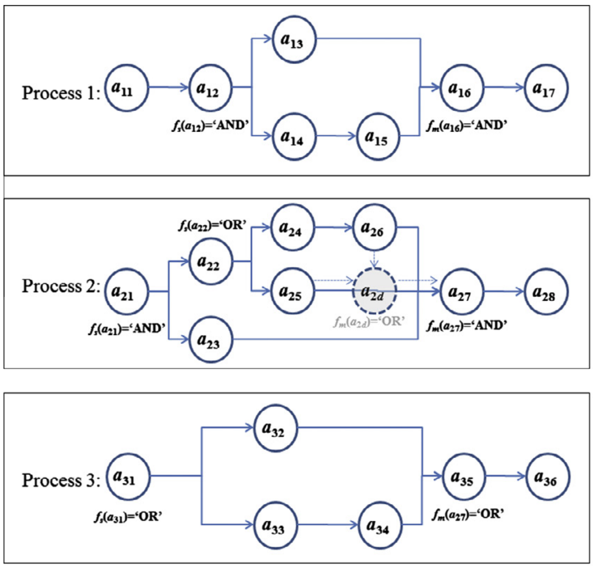
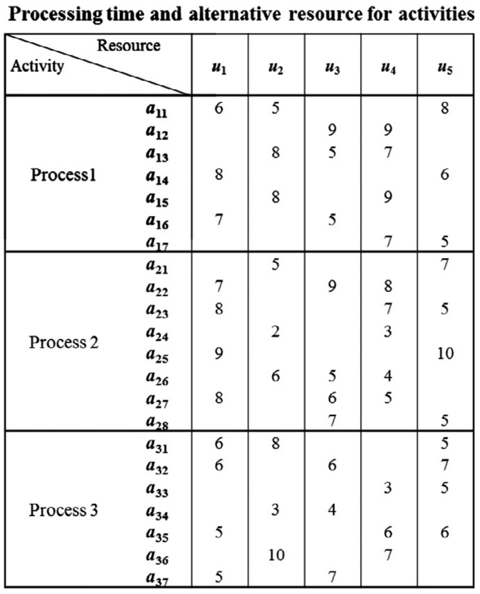

Sequence priority
Resource assignment
$a_{11}, a_{12}, \cdots, a_{1J_1}$
$\cdots$
$a_{I1}, a_{I2}, \cdots, a_{IJ_I}$
$a_{11}, a_{12}, \cdots, a_{1J_1}$
$\cdots$
$a_{I1}, a_{I2}, \cdots, a_{IJ_I}$

Fig. 2. Structure of chromosomes.

```
Step (0)  Let $\bar{S}$ include all activities and $S$ be $\phi$.
          For all $i$ and $j$, initialize $P(a_{ij})$.
Step (1)  From $\bar{S}$, remove the activities $a_{ij}$ such that $P(a_{ij}) = \phi$, and add them to $S$.
Step (2)  If $S = \phi$, then stop.
          Otherwise, select an activity $a_{ik}$ from S, for which $v_{ik} = \min_{a_{ij} \in S}\{\overline{v_{ij}}\}$. To break ties, use one of the following rules to make a selection.
             (i) the activity that comes first to $S$,
            (ii) the activity that has the minimum processing time, or
           (iii) the activity with a higher (or lower) job and operation numbers.
Step (3)  Schedule $a_{ik}$ according to the resource assignment value $\underline{v}_{ik}$.
          Remove $a_{ik}$ from $S$. Calculate the flow time $(F_{ik})$ to this point and update $Rr_k$ and $As_{ik^+}(\forall k^+, (a_{ik}, a_{ik^+}) \in L_i)$
Step (4)  If $|S_{ik}| > 1$,
          If $f_s(a_{ik})$='$AND$', then remove $a_{ik}$ from $P(a_{ik^+})$ for all $k^+$ such that $(a_{ik}, a_{ik^+}) \in L_i$.
          Else, if $f_s(a_{ik})$='$OR$', select an $a_{ik^+}$ that satisfies one of the following conditions:
             (i) the activity that has the minimum sequence chromosome value $(\overline{v_{ik^+}})$ among activities that satisfy
                 $(a_{ik}, a_{ik^+}) \in L_i$,
            (ii) the activity that has the earliest completion time, or
           (iii) the activity with a higher (or lower) operation number.
          Remove $a_{ik}$ from $P(a_{ik^+})$.
          For all $a_{ik^+}$ that have not been selected, call PATH_REMOVE $(a_{ik^+}, w(a_{ik}))$.
          Otherwise if $|S_{ik}| = 1$,
          remove $a_{ik}$ from $P(a_{ik^+})$, for $k^+$ such that $(a_{ik}, a_{ik^+}) \in L_i$.
          Go to Step 1.
```

```
PATH_REMOVE $(a_{ik^+}, w(a_{ik}))$
A temporary activity $a_{ic} \leftarrow a_{ik^+}$
Remove $a_{ik^+}$ from $\bar{S}$
For all $a_{ic^+}$ such that $(a_{ic}, a_{ic^+}) \in L_i$
  Remove $a_{ic}$ from $P(a_{ic^+})$
  If $w(a_{ic^+}) < w(a_{ik})$ and $P(a_{ic}) = \phi$
    $a_{ic} \leftarrow a_{ic^+}$
    PATH_REMOVE $a_{ic}$, $w(a_{ik})$
```





Processing time and alternative resource for activities

Resource / Activity		u_1	u_2	u_3	u_4	u_5
Process1	a_{11}	6	5			8
	a_{12}			9	9	
	a_{13}		8	5	7	
	a_{14}	8				6
	a_{15}		8		9	
	a_{16}	7		5		
	a_{17}				7	5
Process 2	a_{21}		5			7
	a_{22}	7		9	8	
	a_{23}	8			7	5
	a_{24}		2		3	
	a_{25}	9				10
	a_{26}		6	5	4	
	a_{27}	8		6	5	
	a_{28}			7		5
Process 3	a_{31}	6	8			5
	a_{32}	6		6		7
	a_{33}				3	5
	a_{34}		3	4		
	a_{35}	5			6	6
	a_{36}		10		7	
	a_{37}	5		7		

Fig. 3. Process examples for GA.

	a_{11}	a_{12}	a_{13}	a_{14}	a_{15}	a_{16}	a_{17}	a_{21}	a_{22}	a_{23}	a_{24}	a_{25}	a_{26}	a_{27}	a_{28}	a_{31}	a_{32}	a_{33}	a_{34}	a_{35}	a_{36}	a_{37}
Sequence priority part	2	3	4	8	18	18	18	3	13	8	5	20	1	7	4	3	19	10	10	11	17	18
Resource assignment part	1	3	3	5	2	3	5	2	1	5	2	1	2	3	5	5	3	4	2	4	4	1

Fig. 4. An example chromosome.

To elucidate the fitness evaluation, we developed the three-process examples shown on the left side of Fig. 3. The processing times and alternative resources are listed on the right side of the figure. In Process 2, at activity a_{27}, which is a merge point for both a_{21} and a_{22} (which have different split types 'AND' and 'OR'), two different merge semantics are implied. In our notation, we cannot represent two semantics with a single f_m function. For this reason, a dummy node, $a_{2d,}$ was added.

Consider the individual depicted in Fig. 4. The fitness evaluation procedure for that individual, according to the CDP, is outlined below:

Step (0) $\bar{S} = \{a_{11}, a_{12}, \ldots, a_{17,}a_{21}, \ldots a_{28}, a_{31}, \ldots, a_{37}\}$, $S = \phi$

Step (1) $P(a_{11}) = P(a_{21}) = P(a_{31}) = \phi$. Set a_{11}, a_{21}, $a_{31} \in S$ and remove a_{11}, a_{21}, a_{31} from $\bar{S}$.

Step (2) $S = \{a_{11}, a_{21}, a_{31}\}$. Because $\min\{\underline{v}_{11}, \underline{v}_{21}, \underline{v}_{31}\} = \min\{2, 3, 3\} = 2$, a_{11} is selected.

Step (3) Because $\underline{v}_{11}$ is 1, a_{11} is assigned to resource u_1.
$F_{11} = 6$, $As_{12} = 6$, $Rr_1 = 6$.
After a_{11} is removed, S becomes $\{a_{21}, a_{31}\}$.

Step (4) Remove a_{11} from $P(a_{12})$.

Step (1) Because$P(12) = \phi$, $S = S \bigcup \{a_{12}\}$. Remove a_{12} from $\bar{S}$.

Step (2) $S = \{a_{21}, a_{31}, a_{12}\}$, and $\min\{v_{21}, v_{31}, v_{12}\} = \min\{3, 3, 3\} = 3$. Therefore, there is a tie. In the first tie breaking rule, v_{21} and v_{31} are considered. They have the same processing time. Therefore, we selected a_{31} using the tie-break rule.

Step (3) a_{31} is assigned to resource u_5. $F_{31} = 5$, $As_{32} = As_{33} = 5$, $Rr_5 = 5$.

Step (4) $f_s(a_{31}) = 'OR'$. The successors of a_{31} are $\{a_{32}, a_{33}\}$, and
$m = \min\{v_{32}, v_{33}\} = \min\{19, 10\} = 10$.
Select a_{33} and remove a_{31} from $P(a_{33})$.
Remove a_{32} from $P(a_{35})$.
a_{32} is removed from $\bar{S}$ by calling***PATH_REMOVE***$(a_{33}, w(a_{31}))$.

Step (1) $P(a_{33}) = \phi$. Set $S = S \cup \{a_{33}\}$ and remove a_{33} from $\bar{S}$.

Step (2) $S = \{a_{21}, a_{12}, a_{33}\}$ and $\min\{v_{21}, v_{12}, v_{33}\} = \min\{3, 3, 10\} = 3$.

Here, v_{21} and v_{12} constitute times. We select a_{21} because it comes earlier in S than a_{12} does.

These steps are repeated until $S = \phi$. The final result is shown in Fig. 5. The result is an optimal solution provided by the MIP:

$$\sum_{i=1}^{3} F_i = 100, \quad F_{17} = 39, \quad F_{28} = 32, \quad F_{37} = 29$$

4.2. GA procedure

In this section, we present a GA approach to finding a BP execution plan that enhances performance. The use of a GA in BP optimization is not different from that applied to scheduling problems in manufacturing domains. The functions of the proposed GA is illustrated in Fig. 6 and subsequently explained.

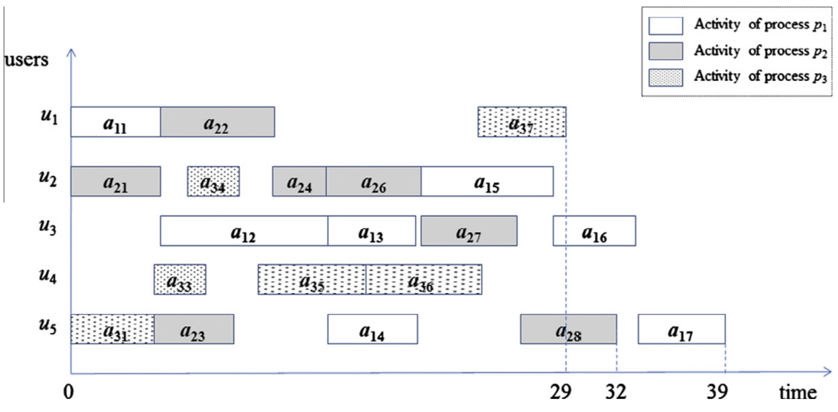


Fig. 5. Gantt chart for the individual in Fig. 4.

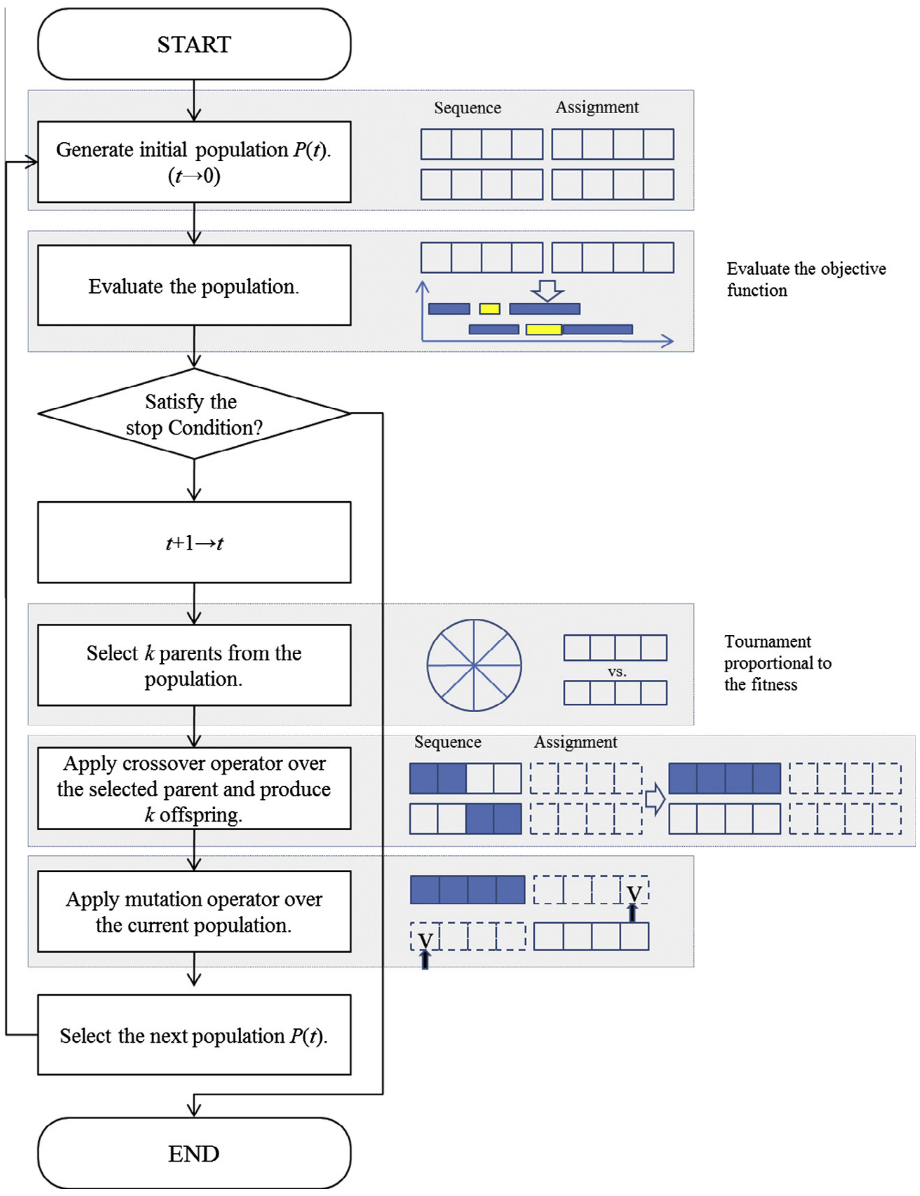

START
Generate initial population P(t). (t→0)
Sequence
Assignment
Evaluate the population.
Evaluate the objective function
Satisfy the stop Condition?
t+1→t
Select k parents from the population.
vs.
Tournament proportional to the fitness
Apply crossover operator over the selected parent and produce k offspring.
Apply mutation operator over the current population.
V
Select the next population P(t).
END

Fig. 6. Structure of the GA.

The following key descriptions clarify the GA.

(1) Representation: An individual consists of two parts: sequence priority and resource assignment. Based on the representation of each individual, a GA provides a method for finding the processing sequence of activities as well as the resource assignment of each activity.
(2) Encoding: The sequence priority of an individual includes as many genes as the total number of activities in a system, and genes are coded with integer values within a predefined range. The genes in the resource assignment part are also coded with integer values representing the resource indices. The length of a chromosome is double the total number of activities.
(3) Crossover: The crossover operates on two parent chromosomes at a time and generates two offspring by combining the features of both chromosomes. For sequence priority, we use 1-point and 2-point crossovers, a partially mapped crossover, and order crossover. For resource assignment, we apply only 1-point or 2-point and uniform crossover operators.
(4) Mutation: The mutation of the sequence priority is conducted by swapping the alleles of two randomly selected genes. Another mutation of the sequence priority of activities is an increasing or decreasing allele of the randomly selected gene. The mutation of the resource assignment is for the purpose of changing the currently assigned resource into one of its alternatives for a randomly selected gene.
(5) Evaluation of fitness: The decoding of the chromosome progresses according to the CDP. The fitness of an individual is evaluated by the objective function as measured by mean flow time, makespan, mean lateness, and mean tardiness.

(6) Stopping condition: The predefined number of generations is reached.
(7) Selection: The fitness proportional tournament method has been employed to generate the new population. That is, an individual with a larger fitness value has a higher probability of survival in the tournament.

5. Computational experiments

In this section, performances of the GA and MIP for BP execution plans are evaluated for small (defined as comprised of a few simple activities) and large (defined as many complex processes) problems. The first experiment (E1) is appropriate for evaluating the mean flow time and makespan of small BPs solved by both MIP and GA. Randomly generated processes between 3 to 13 activities were tested with the parameters listed in Table 2.

The parameters of the GA in E1 are listed in Table 3. In each problem, the fitness value of the GA is the sum of the mean flow time and the makespan.

The experiments of E1 are tested on a computer with Intel® CoreTM2 Quad CPU Q8400 at 2.66 GHz and 2 GB memory. Table 4 shows the results of the makespan and mean flow times by MIP and GA. Optimal solutions by MIP can be obtained only for processes with few activities (12 or fewer). When 13 or more activities are considered, the computation time requires more than 30 h. This means that MIP cannot be used to obtain the optimum solutions in real BP environments where the number of processes typically exceeds 13. Although GA cannot guarantee the optimality, it can generate satisfactory solutions for large BP execution plans.

To compare the quality of MIP and GA more precisely, we computed the relative differences with respect to the GA and MIP approaches. The relative differences of the mean flow time and makespan are defined as $dev_{PF}(\%) = (PF_{GA} - PF_{MIP}) \times$

Table 2
Parameters of business processes for E1.

Parameter	Value
Number of processes	2
Number of resources	5–6
Number of alternative resources for each activity	2
Ratio of ‘AND’ to ‘OR’ splits (merges)	50:50
Processing times	5–10

Table 3
Parameters of the GA in E1.

Parameter	Value
Generations	500
Population size	30
Selection	Fitness proportional
Crossover	PMX
Crossover rate	80%
Mutation	Swapping
Mutation rate	1%

Table 4
Performance of MIP and GA.

Number of activities	Activity Mean flow time					Makespan				
	MIP		GA		Difference	MIP		GA		Difference
	Value	CPU time (s)	Value	CPU time (s)	dev_{PF} (%)	Value	CPU time (s)	Value	CPU time (s)	dev_{PM} (%)
3	28	1	28	0.2	0.00	16	1	16	0.8	0.00
4	43	1	43	0.2	0.00	22	1	22	1.2	0.00
5	40	2	40	1.2	0.00	22	1	22	1.2	0.00
6	70	5	70	1.6	0.00	38	4	38	1.7	0.00
7	84	5	84	2	0.00	43	3	43	2	0.00
8	83	5	83	2	0.00	45	4	45	2	0.00
9	110	33,060	111	2.5	0.91	61	25,220	61	3	0.00
10	100	58,963	102	2.5	2.00	49	33,533	50	2.5	2.04
11	87	72,731	87	2.5	0.00	50	66,015	51	3	2.00
12	87	90,016	90	3	3.45	73	80,716	75	3.4	2.74
13	NA	>100,000	112	3.2	NA	NA	>100,000	73	3.7	NA

$100/PF_{MIP}$ and $dev_{PM}(\%) = (PM_{GA} - PM_{MIP}) \times 100/PM_{MIP}$, respectively, where PF_{GA} and PM_{GA} represent the mean flow time and makespan under the GA, and PF_{MIP} and PM_{MIP} represent the optimum mean flow time and makespan acquired by the MIP formulation. The result of E1 is summarized in Table 4.

We can verify that the results for the mean flow time and makespan of individual processes, up to 8 activities, are exactly the same under the GA and MIP approaches. For problems with more than 8 activities, the GA could not achieve optimal solutions within 500 generations, but the differences in results were insignificant. Therefore, we can confirm that the GA yields a satisfactory BP execution plan.

Because MIP cannot be used to obtain the solution for the problem with the number of activities exceeding 12, we have the necessity of using GA for large BP execution plans. Before we apply GA to larger problems, we conducted a sensitivity analysis to get GA parameters that can provide better solutions for the problems. As a result, we came to discover that the best mean flow time in average can be obtained with a crossover rate of 80% and a mutation rate of 1%. The best makespan in average can be obtained with a crossover rate of 90% and a mutation rate of 1% (See Table 5). The numbers in bold represent the best mean flow time and makespan for the given number of activities.

To overcome the shortcomings of MIP as shown in E1, we tested the performance of the GA in obtaining the mean flow time and makespan for large problems. In E2, we conducted the experiment under the same environment with E1 with the crossover and mutation rates that we obtained from the sensitivity analysis. The 89 problems, with activities ranging from 12 to 100, were generated with the parameters listed in Table 6. In real BP models, we believe that 100 is the typical

Table 5
GA result with different parameter settings.

Crossover rate (%)	Mutation rate (%)	Number of activities						AVG
		15	16	17	18	19	20	
Mean flow time								
70	1	276.2	246.2	271.0	263.4	270.2	**239.2**	261.0
	2	276.4	247.2	**267.0**	260.4	261.6	245.6	259.7
	3	274.2	244.6	273.2	257.6	265.0	248.2	260.5
	4	274.4	246.8	271.4	262.4	263.8	247.8	261.1
	5	278.2	246.8	271.2	257.8	269.8	249.6	262.2
80	1	274.0	245.0	270.8	263.2	**258.8**	240.6	**258.7**
	2	278.4	244.4	272.2	**255.4**	259.8	243.0	258.9
	3	276.2	247.2	272.2	258.6	267.2	247.4	261.5
	4	277.0	246.8	269.6	263.0	270.8	250.4	262.9
	5	275.8	246.4	270.6	263.0	271.2	250.2	262.9
90	1	**272.6**	**243.2**	275.4	261.6	262.0	244.6	259.9
	2	275.0	247.8	269.8	262.4	267.0	246.0	261.3
	3	273.0	246.0	271.6	261.0	269.2	246.0	261.1
	4	276.4	248.4	273.6	262.8	270.4	249.2	263.5
	5	279.0	246.4	271.6	260.2	274.0	249.0	263.4
Makespan								
70	1	94.4	103.6	95.0	90.4	113.2	86.6	97.2
	2	94.8	104.8	96.8	87.6	112.8	88.4	97.5
	3	96.0	104.0	95.6	87.2	113.0	89.2	97.5
	4	93.6	104.4	96.8	88.2	113.0	88.6	97.4
	5	96.6	104.2	97.2	87.4	112.8	90.6	98.1
80	1	93.8	**103.6**	97.6	91.8	114.4	**86.4**	97.9
	2	95.6	104.0	**94.6**	88.2	**111.4**	89.2	97.2
	3	95.0	104.2	95.8	87.2	112.6	90.2	97.5
	4	96.6	104.0	96.0	87.6	113.4	91.4	98.2
	5	95.4	104.6	97.0	87.8	113.6	90.8	98.2
90	1	**93.2**	103.8	96.6	86.8	114.0	87.0	**96.9**
	2	93.8	103.4	97.0	**86.6**	113.0	88.4	97.0
	3	94.2	103.8	95.6	87.0	112.6	91.2	97.4
	4	95.0	104.6	97.4	87.2	113.2	90.6	98.0
	5	96.4	104.2	97.0	88.6	113.0	90.8	98.3

Table 6
Parameters of business processes for E2.

Parameter	Value
Number of processes	3
Number of resources	5–10
Number of activities	12–100
Number of alternative resources for each activity	3
Ratio of 'AND' and 'OR' split (merge)	50:50
Processing times	5–10

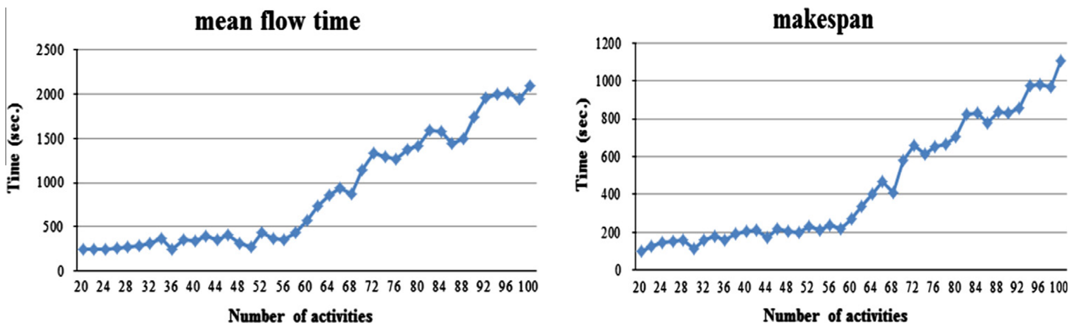


Fig. 7. Performances of GA for mean flow time and makespan.

maximum number used in a single process. If the number of activities in a single process exceeds 100, we recommend dividing the activities into separate processes.

In our experiments, we conducted an experiment for each problem for 500 generations with the population size of 50. The average values of the mean flow time and makespan are selected after 30 runs, because there is no guarantee that the optimal solution can be found with the GA, even after 500 generations.

The results on the mean flow time and makespan of the sample problems using the GA are shown in Fig. 7. From the E2 results, we can conclude that BP execution plans, with respect to the mean flow time and makespan, can be achieved by the GA even for large problems.

6. Conclusions

In this paper, we proposed a new method for establishing BP execution plans. Until recently, developing a systematic BP execution plan has been difficult due to the variableness of human resources and the complexity in BP structures, especially when compared to manufacturing processes. The present research developed a mathematical BP execution model based on a traditional optimization model primarily utilized in manufacturing processes. To that end, we defined a formal BP model and provided MIP formulations for BP optimization. Based on the formulations, we also provided a GA to obtain solutions for processes with many activities. Using the methods obtained in this research, we showed that we can obtain BP execution plans, which can improve BP efficiency by evaluating various measures such as flow time, makespan, and lateness. To show this, we conducted experiments that can be executed by a BPMS using randomly generated BP models. For simple processes, we can optimize a BP execution plan by solving the MIP formulation. However, the MIP is impractical for use on many activities because of the significant length of time it takes to obtain optimal plans. Through our experiments, we showed that a GA approach can provide a satisfactory BP execution plan for large processes.

This research represents a starting point for optimizing BP execution plans; further research remains to be addressed. First, the diversity of BP execution environments, including various BP patterns, must be considered by those preparing formulations. Second, heuristic means of problem solving and method verification need to be developed. Third, a methodology for assigning tasks to groups of users at the design stage and binding a user during execution, commonly used by commercial BPMSs, must be compared with the model proposed in this paper.

This paper dealt with a resource scheduling problem in the field of BP execution plan, and it is new to introduce an optimization technique using a mixed integer program and a meta-heuristic algorithm. We used a genetic algorithm which has been widely used in the scheduling research for a long time because it is the most basic and popular algorithm in meta-heuristics and easy to implement as an initial stage research in this area. Furthermore, the design of chromosomes can be used in other evolutionary algorithms such as tabu search. From numerical experiments, we confirmed that the performance of the genetic algorithm is quite competitive. However, it might be worthwhile to develop hybrid genetic algorithms and other meta-heuristics such as tabu search and ant colony optimization, and we left these challenges as future research directions. Our genetic algorithm can be used as a good benchmarking algorithm for other algorithms which might be developed in the future.

Acknowledgements

The authors are grateful for the useful comments from an associate editor and three anonymous referees. This study was supported by the National Research Foundation of Korea (NRF) grants funded by the Korea Government (MEST) (Nos. 2012R1A2A2A01012355 and 2012R1A1A2008335).

References

[1] I.B. Arpinar, U. Halici, S. Arpinar, A. Dogac, Formalization of workflows and correctness issues in the presence of concurrency, Distrib. Parallel Database 7 (2) (1999) 199–248.

[2] L. Backaker, J.T. Krasemann, Trip plan generation using optimization: a benchmark of freight routing and scheduling policies within the carload service segment, J. Rail Transport Plan. Manage. 2 (1–2) (2012) 1–13.
[3] J. Bae, H. Bae, S.H. Kang, Y. Kim, Automatic control of workflow process using ECA rules, IEEE Trans. Knowl. Data Eng. 16 (8) (2004) 1010–1023.
[4] H. Bae, W. Hu, W.S. Yoo, B.K. Kwak, Y. Kim, Y.T. Park, Document configuration control processes captured in a workflow, Comput. Indus. 53 (2) (2004) 117–131.
[5] G. Baggio, J. Wainer, C. Ellis, Applying scheduling techniques to minimize the number of late jobs in workflow systems, in: The 2004 ACM Symposium on Applied Computing, Nicosia, Cyprus, 2004, pp. 1396–1403.
[6] K. Baker, G. Scudder, Sequencing with earliness and tardiness penalties: a review, Oper. Res. 38 (1) (1990) 22–36.
[7] A. Basu, A. Kumar, Research commentary: workflow management systems in e-business, Inform. Syst. Res. 13 (1) (2002) 1–14.
[8] J.A. Buzacott, Commonalities in reengineered business processes: models and issues, Manage. Sci. 42 (5) (1996) 768–782.
[9] D.H. Chang, J.H. Son, M.H. Kim, Critical path identification in the context of a workflow, Inform. Softw. Eng. 44 (7) (2002) 405–417.
[10] C. Cho, S. Lee, A study on process evaluation and selection model for business process management, Expert Syst. Appl. 38 (5) (2011) 6339–6350.
[11] T. Davenport, The new industrial engineering: IT and business process redesign, Sloan Manage. Rev. (1990) 11–27.
[12] J. Eder, H. Pichler, W. Gruber, M. Ninaus, Personal schedules for workflow systems, Lecture Notes Comput. Sci. 2678 (2003) 216–231.
[13] G. Gharooni-fard, F. Moein-darbari, H. Deldari, A. Morvaridi, Scheduling of scientific workflows using a chaos-genetic algorithm, Proc. Comput. Sci. 1 (1) (2010) 1445–1454.
[14] B.H. Ha, J. Bae, Y.T. Park, S.H. Kang, Development of process execution rules for workload balancing on agents, Data Knowl. Eng. 56 (1) (2006) 64–84.
[15] M. Hammer, The Agenda: What Every Business Must Do to Dominate the Decade, Crown Business, New York, 2001.
[16] K.H. Han, J.G. Kang, M. Song, Two-stage process analysis using the process-based performance measurement framework and business process simulation, Expert Syst. Appl. 36 (3) (2009) 7080–7086.
[17] I. Hofacker, R. Vetschera, Algorithmical approaches to business process design, Comput. Oper. Res. 28 (13) (2001) 1253–1275.
[18] D. Holingsworth, The workflow reference model, Workflow Management Coalition Specification TC00–1003, 1995.
[19] C.-C. Hsu, K.-C. Huang, F.-J. Wang, Online scheduling of workflow applications in grid environments, Future Gener. Comput. Syst. 27 (6) (2011) 860–870.
[20] Z. Huang, X. Lu, H. Duan, Resource behaviour measure and application in business process management, Expert Syst. Appl. 39 (7) (2012) 6458–6468.
[21] J. Hutchison, K. Leong, D. Snyder, P. Ward, Scheduling approaches for random job shop flexible manufacturing systems, Int. J. Product. Res. 29 (5) (1991) 1053–1067.
[22] S. Julia, F.F. Oliveira, R. Valette, Real time scheduling of workflow management systems based on a *p*-time Petri net model with hybrid resource, Simul. Modell. Practice Theory 16 (4) (2008) 462–482.
[23] Y. Kim, S. Kang, D. Kim, J. Bae, K. Ju, WW-flow: web-based workflow management with runtime encapsulation, IEEE Internet Comput. 4 (3) (2000) 55–64.
[24] R.K.L. Ko, S.S.G. Lee, E.W. Lee, Business process management (BPM) standards: a survey, Business Process Manage. J. 15 (5) (2009) 744–791.
[25] A. Kumar, W.M.P.van der. Aalst, E.M.W. Verbeek, Dynamic work distribution in workflow management systems: how to balance quality and performance?, J MIS 18 (3) (2001-2002) 157–193.
[26] A. Kumar, L. Zhao, Dynamic routing and operational controls in a workflow management system, Manage. Sci. 45 (2) (1999) 253–272.
[27] G. Luh, C. Chueh, A multi-modal immune algorithm for the job-shop scheduling, Inform. Sci. 179 (2009) 1516–1532.
[28] G. Martinez, E. Heymann, M. Senar, Integrating scheduling policies into workflow engines, Proc. Comput. Sci. 1 (1) (2010) 2743–2752.
[29] N. Nasr, E. Elsayed, Job shop scheduling with alternative machines, Int. J. Product. Res. 28 (9) (1990) 1595–1609.
[30] J. Oh, N. Cho, H. Kim, Y. Min, S. Kang, Dynamic execution planning for reliable collaborative business processes, Inform. Sci. 181 (2011) 351–361.
[31] H.A. Reijers, W.M.P. van der. Aalst, The effectiveness of workflow management systems: predictions and lessons learned, Int. J. Inform. Manage. 56 (5) (2005) 457–471.
[32] S.H. Rhee, H. Bae, Y. Seo, Efficient workflow management through the introduction of TOC concepts, in: The 8th Annual International Conference on Industrial Engineering Theory, Applications and Practice, USA, 2003.
[33] S.H. Rhee, H. Bae, Y. Kim, A dispatching rule for efficient workflow, Concurr. Eng. – Res. Appl. 12 (4) (2004) 305–318.
[34] S.H. Rhee, H. Bae, N.W. Cho, A more comprehensive approach to enhancing business process efficiency, Lecture Notes Comput. Sci. 4558 (2007) 1611–3349.
[35] P. Senkul, I.H. Toroslu, An architecture for workflow scheduling under resource allocation constraints, Inform. Syst. 30 (5) (2005) 399–422.
[36] H. Smith, P. Fingar, Business Process Management – The Third Wave, Meghan–Kiffer Press, Florida, 2003.
[37] R. Tavakkoli-Moghaddam, A. Rahimi-Vahed, A. Mirzaei, A hybrid multi-objective immune algorithm for a flowshop scheduling problem with bi-objectives: weighted mean completion time and weighted mean, Inform. Sci. 177 (2007) 5072–5090.
[38] V. Valls, A. Perez, S. Quintanilla, Skilled workforce scheduling in service centres, Eur. J. Oper. Res. 193 (3) (2009) 791–804.
[39] X. Wang, C.S. Yeo, R. Buyya, J. Su, Optimizing the makespan and reliability for workflow applications with reputation and a look-ahead genetic algorithm, Future Gener. Comput. Syst. 27 (8) (2011) 1124–1134.
[40] WfMC, 'Interface 1: Process Definition Interchange Process Model', Workflow Management Coalition Specification TC-1016–P, 1998.
[41] Z. Xiao, Z.A. Ming, method of workflow scheduling based on colored Petri nets, Data Knowl. Eng. 70 (2) (2011) 230–247.
[42] Y. Yuan, X. Li, X. Zhu, Deadline division-based heuristic for cost optimization in workflow scheduling, Inform. Sci. 179 (2009) 2562–2575.
[43] R. Zhang, C. Wu, Bottleneck machine identification method based on constraint transformation for job shop scheduling with genetic, Inform. Sci. 188 (2012) 236–252.
[44] J.L. Zhao, E.A. Stohr, Temporal workflow management in a claim handling system, SIGSOFT: Softw. Eng. Notes 24 (2) (1999) 187–195.