

A semi-automatic approach for workflow staff assignment

Yingbo Liu^{a,b,*}, Jianmin Wang^{b,c,d,1}, Yun Yang^{e,2}, Jianguang Sun^{b,c,d,1}

^aDepartment of Computer Science, Tsinghua University, Beijing 100084, China

^bSchool of Software, Tsinghua University, Beijing 100084, China

^cKey Laboratory for Information System Security, Ministry of Education, China

^dTsinghua National Laboratory for Information Science and Technology, Beijing, China

^eSwinburne CITR—Centre for Information Technology Research, Faculty of ICT, Swinburne University of Technology, Australia

Received 8 April 2007; received in revised form 27 November 2007; accepted 17 December 2007

Available online 19 February 2008

Abstract

Staff assignment is of great importance for workflow management systems. In many workflow applications, staff assignment is still performed manually. In this paper, we present a semi-automatic approach intended to reduce the number of manual staff assignment. Our approach applies a machine learning algorithm to the workflow event log to learn various kinds of activities that each actor undertakes. When staff assignment is needed, the classifiers generated by the machine learning technique suggest a suitable actor to undertake the specified activities. With experiments on three enterprises, our approach achieved a fairly accurate recommendation.

© 2008 Elsevier B.V. All rights reserved.

Keywords: Staff assignment; Resource management; Workflow; Machine learning

1. Introduction

In the context of workflow, staff assignment serves for the purpose of specifying the relationship between activities and resources [1]. It ensures that the operation of a workflow conforms to its intended design principles and operates as efficiently and deterministically as possible. In most cases, staff assignment is performed at workflow build-time stage to restrict the range of resources that can undertake an activity, usually by means of the “role” concept. At run-time stage, workflow engine automatically assigns the work to all the resources with that role or to specific resource using some simple mechanisms such as queue lengths or round-robin, etc. [2]. However, in many real situations, such simple run-time

work allocation mechanisms are not sufficient for organizations to correctly assign work to resources.

Consider, for example, an engineering design process in a car manufacturing enterprise we investigated, a typical part design activity is defined to be undertaken by the resources who belong to the designer role, but the actual designer who is responsible for the design can only be specified when a concrete requirement arrives. Because this task is directly related to the manufacturing and eventually determines the quality of the final product, it is unlikely to let designers arbitrarily accept the work items. Therefore, run-time staff assignment is needed. Usually it is performed manually by workflow initiators or monitors. To our knowledge, such manual staff assignment occurs frequently in manufacturing enterprises, especially for those important tasks in business processes.

Although there are various mechanisms of staff assignment in the literature of workflow [2–5], few of them focus on actively recommending resources at run-time, especially using the workflow history information. In Ref. [4], Muehlen envisioned that the workflow history information could be used to improve workflow run-time resource allocation. Russell et al. introduced a pattern (R-HBA), which offers or allocates work items to resources on the basis of their previous execution history, but none of the investigated workflow systems provide

* Corresponding author at: School of Software, Tsinghua University, Zijing, No. 15 Building, 1327A Room, Beijing 100084, China. Tel.: +86 10 51537184; fax: +86 10 62773417.

E-mail addresses: lyb01@mails.tsinghua.edu.cn (Y. Liu), jimwang@tsinghua.edu.cn (J. Wang), yyang@it.swin.edu.au (Y. Yang), sunjg@tsinghua.edu.cn (J. Sun).

¹ Tel.: +86 10 62781776; fax: +86 10 62781776.

² Tel.: +61 3 92148752; fax: +61 3 98190823.

direct support for this pattern [2]. Ly and Rinderle et al. proposed a method to derive staff assignment rules from event logs that mainly aims at facilitating the assignment at build-time stage [5,6].

In this paper, we present an approach intended to reduce the amount of manual staff assignment performed at workflow run-time instantiation and execution stages. Our approach applies a machine learning algorithm to the workflow event log in order to learn various kinds of activities that each actor undertakes. When staff assignment is needed, the classifiers generated by the machine learning technique suggest a suitable actor to undertake the specified activities.

Our approach requires an enterprise's workflow system to have had an event log for some period of time and the corresponding workflow models, so that the patterns of who executes what kinds of activities can be learned. Using our approach, we have been able to correctly suggest appropriate actors to undertake the activities with overall prediction accuracies of 82.88%, 79.48% and 79.44%, respectively in three vehicle manufacturing enterprises.

This paper makes two contributions: firstly, it presents an approach for helping automate workflow staff assignment in workflow management systems; secondly, it evaluates the applicability of a machine learning approach for staff assignment using real world datasets.

This paper is organized as follows: we begin with presenting some background information about the workflow event log and general information about three enterprises (Section 2). Given this background, we describe our semi-automated approach for staff assignment (Section 3) and present the results of applying our approach on real world datasets (Section 4). We then discuss some possible improvements (Section 5). Related works about workflow resources allocation are discussed in Section 6. Finally, we summarize the paper (Section 7).

2. Background

Our work is based on a workflow management system called Tsinghua InfoTech Product Lifecycle Management (TiPLM) workflow. This workflow management system is a part of Tsinghua InfoTech Product Lifecycle Management solution [7], which is a strategic business solution in support of the collaborative creation, management, dissemination, and use of product definition information across the enterprise [8]. By the time this paper is written, TiPLM has been successfully deployed in 38 Chinese manufacturing enterprises.

In this section, we provide a brief introduction of TiPLM workflow as well as some basic concepts of workflow management. Besides, we give an overview on workflow applications and current staff assignment practice in three enterprises.

2.1. Workflow model and event log in TiPLM workflow

In TiPLM workflow, a *workflow* model is described by a directed graph that has four different kinds of nodes:

- *Activity nodes* represent tasks that are undertaken by some actors.
- *Route nodes* represent decision points that determine the execution flow.
- *Start node* denotes the start of a workflow.
- *End node* denotes the end of a workflow.

Nodes are connected by directed links to define different kinds of control flows. There are two kinds of links in TiPLM workflow: normal link and false link. Normal links represent normal execution flows, while false links represent those execution flows when route node conditions are evaluated to be false.

Fig. 1 shows a screenshot of the TiPLM workflow model editor, in which an electronic configuration change process of a car manufacturing enterprises is being edited.³

A workflow model can be instantiated multiple times, each instantiation corresponds to an actual execution of the workflow model, which is called a *workflow* instance. In practice, multiple workflow instances may be concurrently active and they execute without referencing to each other.

When an activity node is instantiated at the execution time of a workflow instance, the TiPLM workflow engine reads the staff assignment of this activity, and puts the *work items* into the assigned actors' worklist. Actors then periodically connect to the worklist via a desktop application, pull the work item assigned and execute it. Once the actor commits the work item, the corresponding activity will be marked as completed and its succeeding activities will be instantiated. Meanwhile, an *event entry* is generated to log the actor's operation, e.g. work item's timestamp, actor's identity, workflow instance information, etc. TiPLM workflow's *event log* stores all these event entries.

Fig. 2 shows a worklist handler of TiPLM workflow, through which an actor can monitor workflow instances, execute/commit work items using pop-up menus and watch the future work items assigned in the same workflow instance.

2.2. Workflow applications in three enterprises

In previous section, we have outlined the functionalities of the TiPLM workflow management system. In order to test the validity of our approach, we collected workflow history data from three manufacturing enterprises. The first enterprise is Xiamen KingLong United Automotive Co. Ltd. [9]; the second one is Hebei Zhongxing Automobile Co. Ltd. [10] and the third one is Datong Electronic Locomotive Co. Ltd. [11]. We investigate them because workflow management systems have been successfully used in many aspects of their business, e.g. engineering design and review, release management, notification dissemination, purchase management, customer service, etc.

³ In order to make it discernable, we repainted the false links using dotted line, however, in real workflow model false links are represented by solid red line.

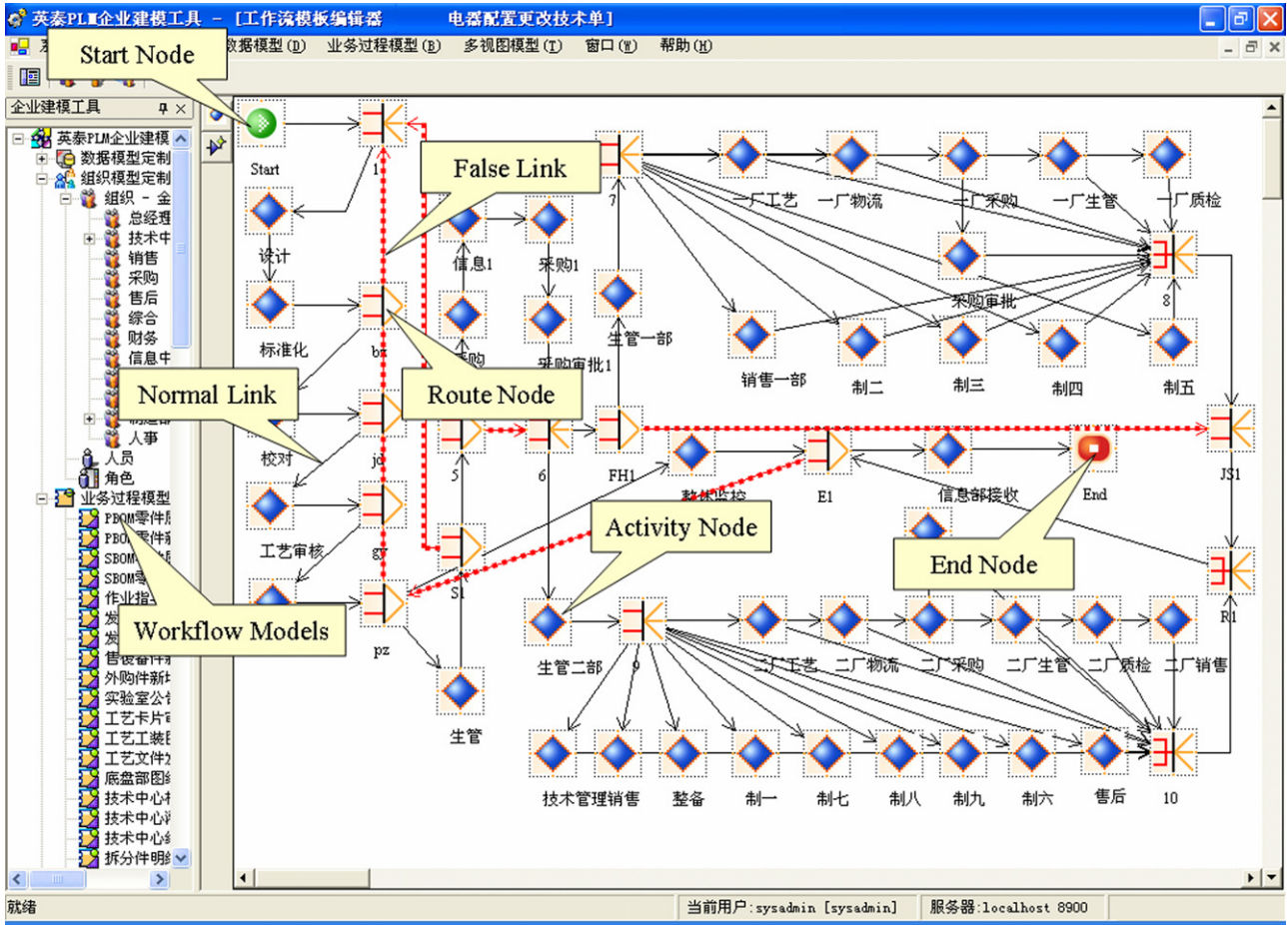


Fig. 1. TiPLM workflow model editor.

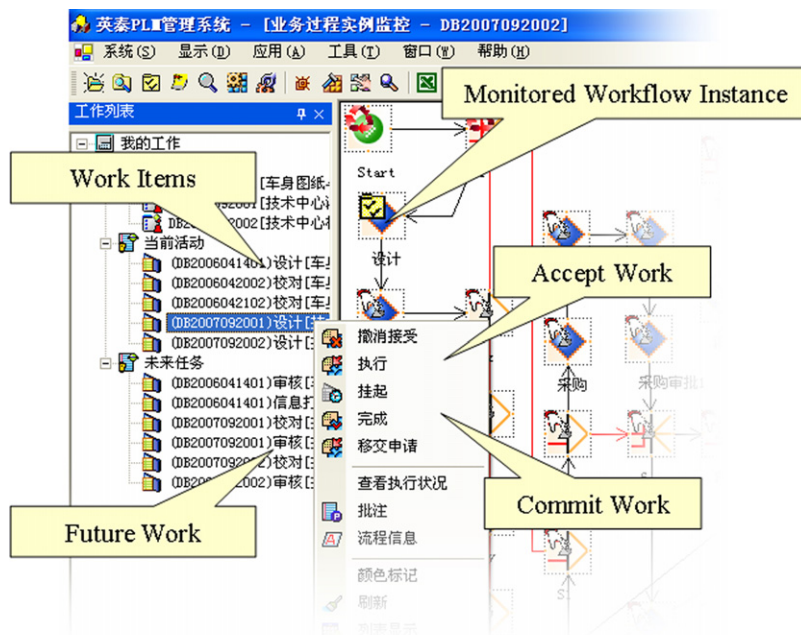


Fig. 2. TiPLM worklist handler.

Table 1
Overview of three enterprises' workflow history data

	Enterprise		
	A	B	C
Operation time (days)	117	421	949
Event entries	10,808	42,099	99,765
Number of actors	179	244	147
Workflow models	21	24	49
Number of activities	256	399	922
Related departments	46	29	28

We collect their workflow history data by writing SQL scripts against their TiPLM database and importing the retrieved data into a separate database for analysis. The scripts extract three kinds of information: workflow history information (workflow instance control data, workflow event log) workflow model definition and organizational information. Table 1 is an overview of workflow history data we collected in three enterprises. For the confidential reason we use A–C to represent these enterprises without disclosing the correspondence between these letters and the enterprises.

As shown in Table 1, in all these enterprises, workflows have operated for certain periods of time and many actors from different departments are involved in the execution of workflows, which is quite suitable for testing our approach.

2.3. Staff assignment practice using TiPLM

The staff assignment practice in these enterprises is quite similar. It is performed in three stages, which are illustrated in Fig. 3. At build-time stage, the workflow designer specifies the range of potential actors for each activity node in the workflow model (step 1 in Fig. 3). At run-time, anyone who wants to

instantiate a business process may pick up a corresponding workflow model from the model list to create a workflow instance, and then, he (or she) manually assigns concrete actors for those important activities in the workflow instance (step 2). After a workflow instance has been activated, the workflow monitor can modify unsuitable staff assignment (step 3).

In this scenario, a considerable amount of staff assignment is performed at the workflow instance level. When the business pressure becomes higher and higher, more and more workflow instances will be created, consequently, workflow initiators or monitors are required to repeatedly perform the task of manual staff assignment, which is tedious and time consuming. Therefore, if we can correctly recommend a suitable actor in step 2 or step 3, and just let workflow initiators or monitors confirm the recommendation, the staff assignment overhead can be reduced for them.

3. A semi-automated approach to workflow staff assignment

Our approach of semi-automatic staff assignment is based on machine learning. Its rationale is depicted in Fig. 4, and can be described as follows: for a given activity, each execution of this activity can be viewed as a training sample and the event entries of those previous activities in the same workflow instance can be viewed as this training sample's features. The training sample may have a label that indicates the actor who has undertaken the activity. A supervised machine learning algorithm takes a set of training samples with known labels as input, and generates a classifier. The generated classifier can then be used to assign a label to an unknown sample, which, in the context of workflow, is an unassigned activity in the workflow instance. The process of creating a classifier from a set of training samples is known as training the classifier.

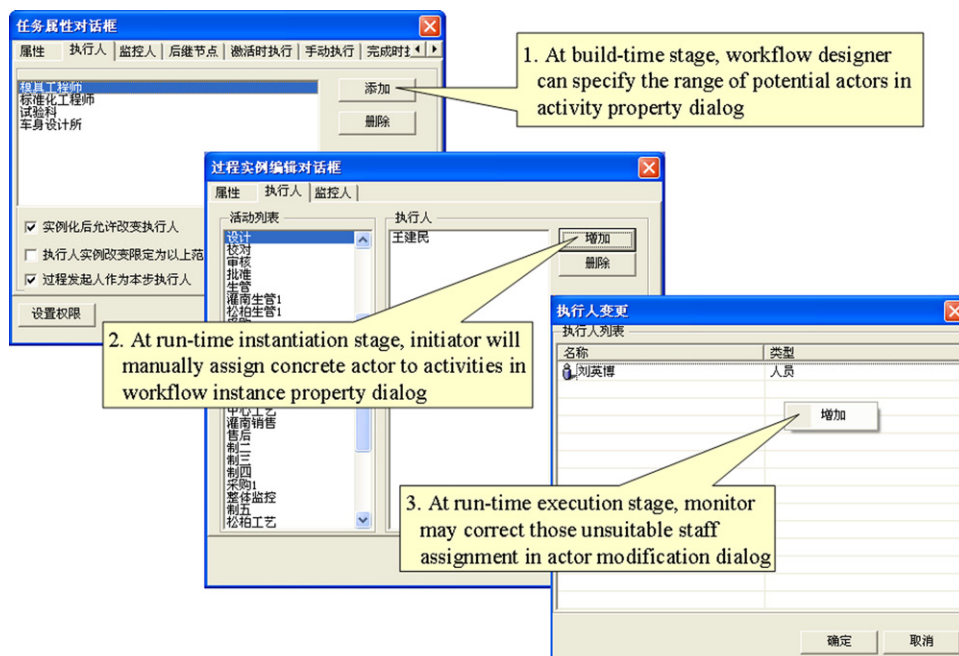


Fig. 3. Staff assignment in different stages.

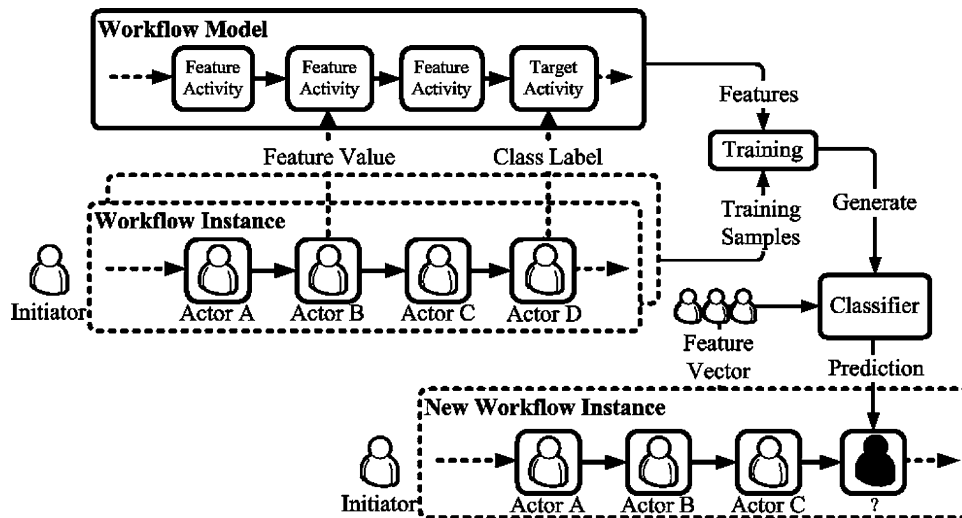


Fig. 4. Rationale of semi-automated workflow staff assignment.

Our approach is semi-automated because the workflow initiator or monitor must decide whether the suggested actor is the actual actor to whom the activity will be assigned, and he (or she) may make this choice based on the knowledge not available in the event log, such as the workloads of actors, or actors' availability, etc.

As is typical in machine learning, we evaluate the performance of each classifier using k -fold cross-validation [12]. For a given training set, we separate them into k mutually exclusive subsets with approximately equal size, and iteratively test each subset using classifier trained by the remaining $k - 1$ subsets. The accuracy is defined by the average ratio of appropriate assignment for k iterations. In the real situation, k is selected according to the characteristics of the training set. In general, 10-fold is recommended for estimating the accuracy due to its relatively low bias and variance. Therefore, we let $k = 10$ in our experiments.

In order to train a classifier, we take the following five steps that are detailed in this section:

- Step 1: Preprocessing the event log.
- Step 2: Selecting target activities.
- Step 3: Finding precedence activities in the workflow model.
- Step 4: Constructing the training set from the event log.
- Step 5: Applying machine learning to obtain the classifier.

3.1. Preprocessing the event log

Before a machine learning approach can be applied, preprocessing is needed to exclude those unsuitable event entries, e.g. unfinished event entries or incomplete event entries, etc. In Refs. [13,14] several preprocessing strategies are discussed. In our experiment, we preprocess the event log by excluding those unfinished and incomplete event entries so as to ensure that the remaining event entries represent actually finished activities and the actor identity information is not missing.

Another important function of preprocessing is to filter out those event entries with inappropriate staff assignment, to

prevent the negative impact from being incorporated into training. However, this preprocessing is not performed in our experiment. We do not perform this processing because assessing the appropriateness of staff assignment requires profound knowledge of management and in depth understanding of enterprises' business. Although there are many staff assignment evaluation methods in the literature of management [15], most of them require intensive manual processing of domain experts, which is not likely to be automated. Therefore, in this paper, we do not tackle the problem of assessing the appropriateness of staff assignment.

3.2. Selecting target activities

For a set of workflow models in an enterprise, it is not necessary to predict the suitable actor for all activities. Firstly, in practice, some activities' staff assignment has already been fixed to specific actor at build-time, so at run-time stages, no extra staff assignment effort is needed for these activities. Secondly, if an activity can only be performed by very few actors, the assignment rule would be so obvious that can be easily derived by initiators or monitors themselves. Finally, if an activity is rarely executed, e.g., those activities designed to handle special conditions, the size of the training set would be rather small, and the staff assignment pattern may not be clearly demonstrated in the training set. Therefore, activities that are suitable to be predicted should be executed for many times, undertaken by many actors and more importantly, manual staff assignment at run-time stages is needed for them.

In our previous work [16], we selected a set of representative activities that strictly met the aforementioned requirements, and we achieved the satisfactory result. However, it just shows the feasibility of applying the machine learning-based approach. In this work, we expand the scope of our investigation to rigidly evaluate the performance. We consider an activity interesting as long as it satisfies the following criteria. Firstly, manual staff assignment at run-time stage is needed for this activity, secondly, more than one actor

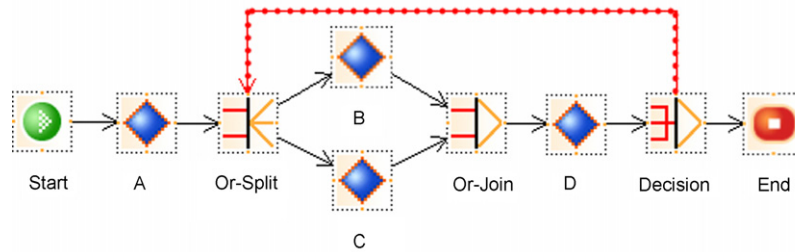


Fig. 5. Example TiPLM workflow model with a loop structure.

undertakes it, and finally its execution number meets the basic requirement of 10-fold cross-validation.

After this process, we selected 35 activities from enterprise A, 68 activities from enterprise B and 125 activities from enterprise C.

3.3. Finding precedence activities in workflow model

After selecting the target activities, the next step is to find out the precedence activities for each target activity. In order to meet this end, we first simplify the workflow models into directed acyclic graphs called workflow control graph.

Definition 1 (Workflow control graph).

A workflow control graph of a workflow model is a directed acyclic graph represented by tuple $W(S, F, A, R, L)$, where S and F are the start and end nodes, respectively, A is a set of activity nodes that are undertaken by different actors, R is a set of router nodes (e.g. choice, merge, fork, join) and L is a set of directed links between these nodes.

It is very important to mention that, in many real situations, loop structure is always defined in workflow models. In TiPLM workflow, loop structure is usually used to redirect the execution of workflow back to the previous activities, so that actors may redo the tasks. In most cases, a false link is defined to realize such redirection. In our experiment, we remove the loop structures with a very simple method: first, we exclude those false links in a workflow model that may lead to the loop structures. Then we manually check the remaining workflow model to ensure that no other loop structures exist and each activity node is located in the paths from the start node to the end node. Because most TiPLM workflow loop structures contain false links, this method works quite efficiently. However, for other kinds of workflow models described in different modeling languages, it would be better to use automatic methods to realize the simplification [17].

Fig. 5 shows an extremely simplified production planning process created using TiPLM workflow, in this process, a purchaser may submit an order to one of the two factories (activity A). Then, the scheduler in the selected factory processes the order, and generates a production plan (activity B or C). After the plan is generated, a production manager reviews the plan, and decides whether it is suitable to be carried out in the workshops (activity D). If it is approved, the workflow executes to an end and the new plan is carried out. Otherwise, a false link is defined to redirect the execution from the ‘decision’ node back to the ‘or-split’ node, so that the scheduler can reschedule the production plan.

In order to simplify this workflow model into a workflow control graph we remove the false link in Fig. 5 and the resulting workflow control graph is depicted in Fig. 6.

Definition 2 (Precedence node sequence).

For a given activity $a \in A$ in a workflow control graph W , the precedence node sequence of a , denoted by $\text{pre}(a)$, is an ordered set of nodes $(n_0, n_1, n_2, \dots, n_k)$ that have a precedence relationship with activity a . The order of each node is defined by a fixed topological order in W that begins at start node S and ends at target activity a .

Note that, in Definition 2, the topological order is just used to maintain a sequential order of nodes whenever $\text{pre}(a)$ is evaluated. It does not matter which topological order is used if there are multiple topological orders.

Moreover, since the sequence is obtained from the workflow control graph, it may contain activity nodes as well as route nodes. In our experiment, we just consider the activity nodes. We do not consider router nodes because, in TiPLM workflow, staff assignment is not performed against route nodes and there is no event entry of route nodes in TiPLM workflow’s event log.

Definition 3 (Precedence activity sequence).

For a given activity $a \in A$ in a workflow control graph W , the precedence activity sequence denoted by $\text{pre}_A(a)$ is an ordered

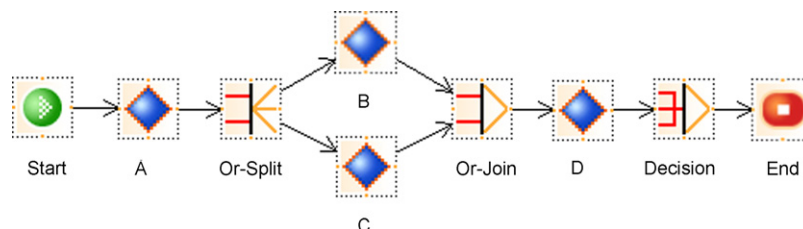


Fig. 6. Workflow control graph after simplification.

set of activities $(a_0, a_1, a_2, \dots, a_k)$ obtained by filtering out all the route nodes of $\text{pre}(a)$ while maintaining the original order of remaining activities.

For example, consider the workflow control graph in Fig. 6, let activity D be the target, there are two topological orders that are suitable for representing the precedence node sequence, i.e., (S, A, or-split, B, C, or-join) and (S, A, or-split, C, B, or-join). In order to maintain the order of nodes, we always keep using a fixed order, e.g., (S, A, or-split, B, C, or-join), whenever $\text{pre}(D)$ is evaluated. Once the precedence node sequence has been determined, the precedence activity sequence of D can be found, namely $\text{pre}_A(D) = (S, A, B, C)$.

3.4. Constructing training set from event log

In order to train a classifier, we need to provide a set of training samples to the machine learning algorithms. In this section, we define the process of constructing the training set by adopting some notions defined in Ref. [18].

Definition 4 (*Workflow instance event log*).

For a given workflow model, let P be the set of actors. $E = A \times P$ is a set of possible event entries (e.g. (a, p) means the execution of a is performed by p). The event log of a workflow instance $C \in E^*$ is a sequence of event entries $(e_0, e_1, e_2, \dots, e_n)$, that describe the execution of workflow.

For convenience, we use $e_0 = (S, p)$ to represent the instantiation event of workflow instance, and we adopt the same operations π_A and π_P defined in Ref. [18] to obtain the actor and activity information from an event entry, namely $\pi_A(e) = a, \pi_P(e) = p$ for some event entry $e = (a, p)$.

Table 2 is an example event log of workflow model depicted in Fig. 5, each row in Table 2 corresponds to one workflow instance's event log. There are six actors $\{P_1-P_6\}$ who have participated the workflow instances C_1-C_4 . Note that, in the event log of C_3 (the third row in Table 2), activities C and D have been iteratively executed twice, thus, there are two event entries of C (e_2, e_4) and D (e_3, e_5) compared to C_1 and C_2 . Similarly, there are also two event entries of activities C and D in C_4 .

As described in the beginning of Section 3, we treat each execution of a target activity as a training sample, in the context of machine learning, a training sample is typically represented by a vector with at least two dimensions, one represents class label and the other(s) represent features. However, there are two issues that need to be addressed before constructing the training samples.

Firstly, because training samples are constructed from the workflow event log, a considerable amount of information can be used to form the vector of a training sample, e.g., activity execution time, actor's identity, etc. It is very important to use the most relevant information with respect to the target problem to form the training sample, otherwise, the learning efficiency and prediction accuracy will be negatively affected by irrelevant features [12].

In our experiments, we use actor identity information of event entries to generate the training samples. We make this selection because this information is generally available in most of the workflow systems [19]. Another important reason for us to make this selection is that the feedback from enterprise users has clearly revealed the fact that different groups of people tend to work together in a workflow instance. Hence, it is reasonable to base our learning on actors' identity information.

Secondly, in many real situations, because of the loop structures in workflow models, some activities may be executed for several times in one workflow instance, which results in multiple event entries of these activities (e.g., the event entries of activities C and D in C_3 in Table 2). Such situation may lead to ambiguity when we are trying to construct a training sample for the execution of the target activity. In order to avoid such ambiguity issue, we only consider the latest event entry before the execution of target activity. In this way, we ensure that each execution of the target activity correspond to a unique training sample.

Definition 5 (*Latest activity performer projection*).

For a given event entry $e_k \in C$ and an activity $a \in A$, the latest activity performer projection \triangleright_a produces the latest actor who has executed activity a in C before e_k .

$$\triangleright_a^k(C) = \begin{cases} \pi_P(e_i), & \exists i \in [0, k], (\pi_A(e_i) \\ = a \wedge \forall j \in (i, k), \pi_A(e_j) \neq a) \\ \text{null}, & \forall i \in [0, k], \pi_A(e_i) \neq a \end{cases} \quad (1)$$

Definition 6 (*Training sample*).

For a given activity $a \in A$ and an event entry of this activity $e_k \in C$, the training sample of e_k consists of a vector with a class label and one or more features that describe the execution of a in C where the class label is represented by $\pi_P(e_k)$ and the feature vector is represented by the latest performers of activities in a 's precedence activity sequence:

$$\text{sample}(e_k) = (\triangleright_{a_0}^k(C), \triangleright_{a_1}^k(C), \dots, \triangleright_{a_n}^k(C), \pi_P(e_k)), \quad \forall i \in [0, n], a_i \in \text{pre}_A(a) \quad (2)$$

For example, consider the event log of C_3 in Table 2, let activity D be the target activity, and (S, A, B, C) be the precedence activity sequence of D. There are two event entries of D, i.e., e_3 and e_5 , which means that activity D has been iteratively executed twice. For the first iteration of D, i.e., e_3 , the latest activity performer of S, A, B and C before e_3 are P_1, P_5 , null and P_2 , so, the training sample of e_3 is $(P_1, P_5, \text{null}, P_2$ and $P_6)$ with P_6 as the class label. For the second iteration e_5 , the latest performer of activity C has been changed to P_3 (as shown

Table 2
Example event log of workflow instances

	e_0	e_1	e_2	e_3	e_4	e_5
C_1	(S, P_1)	(A, P_2)	(B, P_3)	(D, P_4)		
C_2	(S, P_1)	(A, P_3)	(C, P_2)	(D, P_4)		
C_3	(S, P_1)	(A, P_5)	(C, P_2)	(D, P_6)	(C, P_3)	(D, P_4)
C_4	(S, P_1)	(A, P_5)	(B, P_3)	(D, P_6)	(B, P_2)	(D, P_6)

Table 3
Training set of activity D

S	A	B	C	D
P ₁	P ₂	P ₃	Null	P ₄
P ₁	P ₃	Null	P ₂	P ₄
P ₁	P ₅	Null	P ₂	P ₆
P ₁	P ₃	Null	P ₃	P ₄
P ₁	P ₅	P ₃	Null	P ₆
P ₁	P ₅	P ₂	Null	P ₆

in row of C₃ and column of e₄), so the training sample of e₅ is (P₁, P₅, null, P₃ and P₆), whereas the associated class label is still P₆.

Finally, the training set of the target activity is constructed by collecting training samples for all the event entries of the target activity. Table 3 lists the training set of activity D constructed from event entries in Table 2.

3.5. Applying machine learning to obtain classifier

We use three widely accepted machine learning algorithms to obtain a classifier. The first one is C4.5 decision tree [20]. We chose this algorithm because it is proposed by previous researches [5,14] and it resembles the decision process of people. The second one is Naïve Bayes [21], which is a probabilistic machine learning algorithm. We select this algorithm because it is quite simple to realize in the real world applications and it can provide a reference point for

finding algorithms that are more appropriate. The last algorithm is support vector machine (SVM) [22]. We use it because it is suitable for small training sets.

As the purpose of this paper is to demonstrate the applicability of the machine learning approach in workflow staff assignment, we use existing tool WEKA [23] to train the classifiers and perform the tests. To this end, we first write a PL/SQL script to retrieve the training data of selected activities from our aforementioned analytical database. Then we transform the data into file format that is acceptable by WEKA. Finally, we let WEKA automatically perform the training and testing using batch processing.

4. Experiments and evaluation

To demonstrate how well our approach can be used in real world applications, and to determine which algorithm is more suitable for staff assignment, we applied our approach on three enterprises' datasets.

4.1. Overview of datasets

Fig. 7 demonstrates the basic properties of training sets constructed from 228 selected activities. In Fig. 7, each subfigure is numbered by the label located on the top right corner (a–i), each column of subfigures corresponds to one enterprise, and each row of them demonstrates one specific aspect of the property. The horizontal axis of these figures

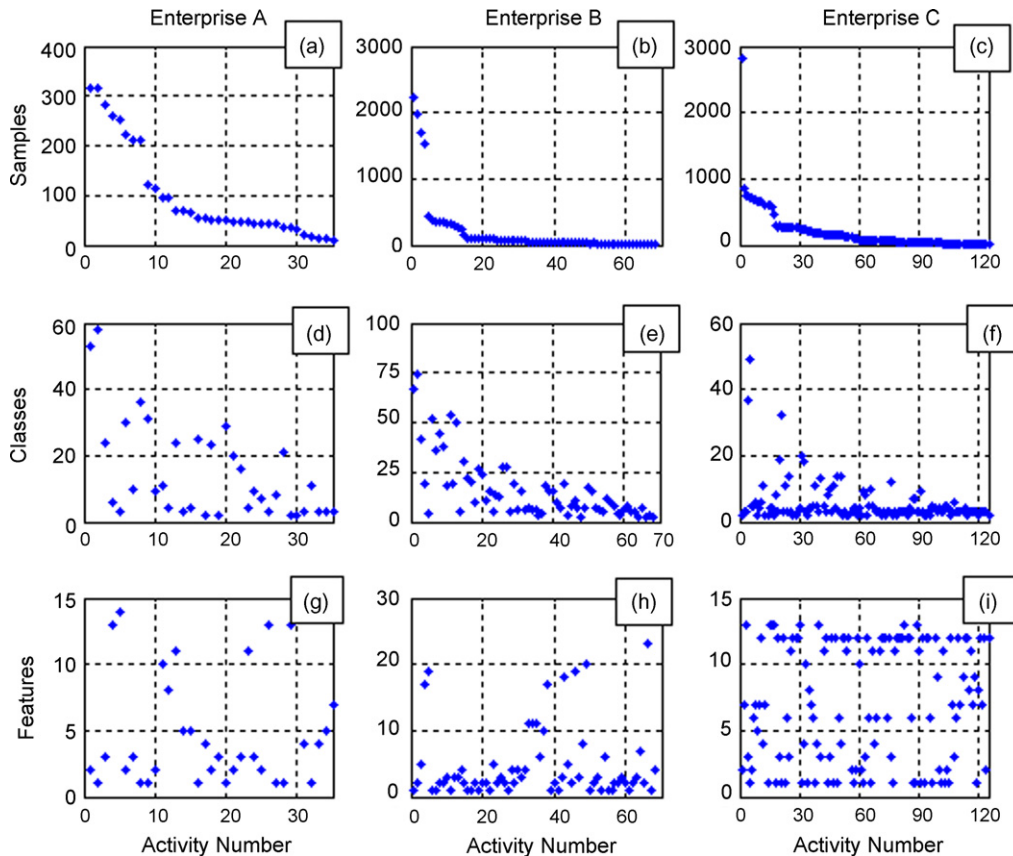


Fig. 7. Basic propriety of selected activities in three enterprises.

represents the activity number, which is defined in decreasing order of their training set sizes.

In Fig. 7, the first row of subfigures (subfigure a–c) illustrates each activity’s training set size. The vertical axis represents the number of training samples for each activity. In the real situation, it reveals the number of times that the target activity has been executed. The second row of subfigures illustrates the number of classes for each training set. The vertical axis represents the class number. In our approach, the class is represented by actor’s identity. Therefore, this number demonstrates how many different actors have executed the target activity. The last row of subfigures depicts the number of features for each training set. The vertical axis is the feature number. Because we use previous activities of the target activity as features, this number demonstrates how many activities in a workflow model have the precedence relationship with the target activity.

For example, in enterprise A (the first column of subfigures in Fig. 7), the property of the first activity corresponds to the left most pixel of each subfigure. This activity is a review activity of a CAD drawing design process, which is the second activity of the workflow model. 58 different actors executed it for 316 times in the past. Therefore, the training set of this activity contains 316 training samples, 58 classes and 2 features (one feature for the actor of the first activity and the other for the initiator of the workflow instance). It is important to mention that, in Definition 3, the start node is also included in the precedence activity sequences to represent the instantiation event of workflow, so there is always one more feature than the number of activities.

According to Fig. 7, we may find that the size of the training set is not the same for each activity, which implies a different execution frequency. However, this unbalanced execution frequency does not have much impact on the number of past actors (i.e., the second row of subfigures), which means the activities are designed to be undertaken by many actors. Moreover, judging from the number of features, we can see that the activities that need run-time staff assignment are located in different places of the workflow model.

4.2. Prediction accuracy of classifiers

Fig. 8 shows the prediction accuracy of 10-fold cross-validation for the selected activities in three enterprises. In order to provide a reference, we connected the pixels of Naïve Bayes classifiers using dotted lines. As shown in the figure, the prediction accuracy is not the same for all the selected

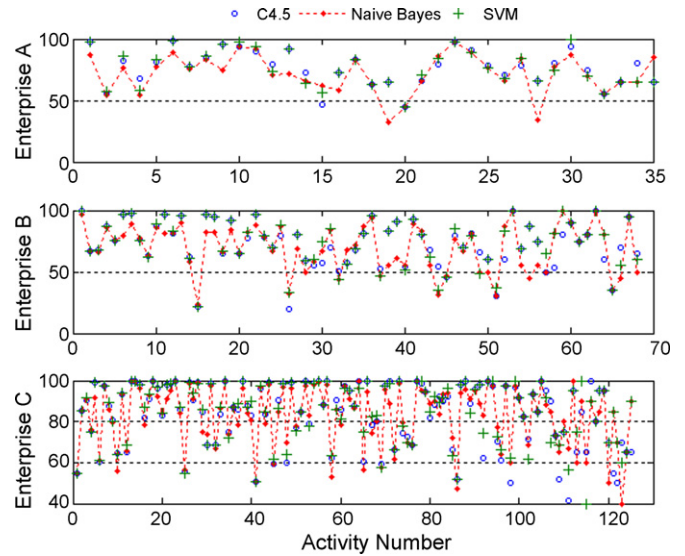


Fig. 8. Prediction accuracy of classifiers in three enterprises.

activities. Some of them are well predicted by the generated classifier (>80%), but some of them are not. However, most classifiers have achieved a prediction accuracy that is greater than 50%. Just few activities’ staff assignment is poorly predicted in three enterprises.

4.3. Overall performance of learning algorithms

Table 4 lists the overall prediction accuracy of three algorithms. We use the percentage of correct predictions as a measure of overall performance. According to the result, in three enterprises, all the algorithms have achieved an overall prediction accuracy of well over 75%. SVM generally performs the best, but only marginally.

Table 5 lists the training and testing time of three algorithms. The testing time of three algorithms is very quick, but the training time differs greatly. Naïve Bayes performs the best in terms of time consumption. In contrast, SVM’s training time is very long.

4.4. Analysis of some special cases

As a further investigation, we evaluated some special cases. Table 6 lists the properties of some worst predicted activities in three enterprises (the first, third and fifth rows). Their related workflow models are depicted in Fig. 9. As a comparison, we also provide some of the best predicted activities in Table 6 (the second, fourth and sixth rows).

Table 4
Overall prediction accuracy in three enterprises

Enterprise	Number of predictions	Number of correct predictions			Prediction accuracy (%)		
		C4.5	Naïve Bayes	SVM	C4.5	Naïve Bayes	SVM
A	3,447	2,739	2,532	2,740	79.460	73.455	79.489
B	13,483	10,669	10,294	10,712	79.129	76.348	79.448
C	24,347	20,028	19,766	20,178	82.261	81.185	82.877

Table 5
Training and testing time of three algorithms

Enterprise	Training time (s)			Testing time (s)		
	C4.5	Naïve Bayes	SVM	C4.5	Naïve Bayes	SVM
A	0.943	0.14	15,134.67	0.05	0.121	3.464
B	18.504	1.242	39,105.74	3.004	1.222	45.524
C	16.90	3.125	10,765.02	2.03	0.98	3.14

In enterprises A and B the worst predicted activity is at the beginning of workflow model. In enterprise A, the activity is a design activity of the technical document review process in the second level assembly. In enterprise B, it is a review activity of the technical notification dissemination process. In the real business, these activities are regulated to be undertaken by process initiators themselves, so the staff assignment pattern should be very easy to understand. However, in their training sets, there are many actors who have executed these activities and most of them have just performed for less than three times. The rest of execution is dominated by only one or two actors, which results in a highly biased distribution of training samples. Therefore, when 10-fold cross-validation is performed, some of the class labels in the testing set are not represented in the training set. Consequently, the prediction accuracy is not good. In these enterprises, such situation usually happens when new workflow models start to operate, because in this period, only a few actors actively engage in the execution of new workflow models, while others rarely take part in.

Also at the beginning of workflow model, well predicted activities do exist, for example, the activities represented by the fourth and sixth rows in Table 6 in enterprises B and C. The difference is that their training samples are sufficient for each class, which means that most actors have performed these activities for sufficient number of times. Therefore, the staff assignment pattern is obvious for the learning algorithm to derive.

In enterprise C, the worst predicted activity is located at the end of the workflow model. This activity is a confirmation activity of the production management department in the car electronic configuration change process. In the real business of enterprise C, this activity is conditionally performed, which leads to a limited number of training samples. Although it has been undertaken by very few actors and there are many features to learn. The staff assignment pattern is not clearly demonstrated in such a small training set, thus the classifier's performance is not good.

4.5. Evaluation based on experiment results

According to the background in Section 2, correct staff assignment prediction implies potential saving of the manual staff assignment effort. Therefore, judging from workflow initiators or monitors' point of view, the overall prediction accuracy reported in our paper has clearly demonstrated the applicability of our approach.

In addition, we have also observed that despite the marginal difference of prediction accuracy for three learning algorithms, the prediction accuracy is also influenced by the quality of the training set, namely, how clearly the staff assignment pattern displays itself in the training set. In the initial running phase of the workflow, the staff assignment pattern is not likely to be clearly represented by the limited number of event entries, so the prediction accuracy might not be satisfactory. However, as time goes by, more and more training samples will be gradually accumulated from the event log, which means that the staff assignment pattern will be clearer and clearer, hence, the prediction accuracy will be better.

Finally, irrespective of the results presented in this paper, it is needed to point out that staff assignment is not just related to workflow initiators or monitors. It also determines an organization's overall performance. In our experiments, we have made no attempt to assess the quality of staff assignment. Our approach learns from past staff assignment, regardless of whether they are successful or not. Consequently, the classifiers trained and evaluated may reflect good practices as well as poor ones. Therefore, further study is needed to let real enterprise users evaluate the quality of automatic staff assignment and we plan one such study as part of our future work.

5. Discussion on improvement

Despite the results we report in this paper, it is still reasonable for us to believe that there is plenty of space for improvement. In this section, we discuss some possible directions.

Machine learning algorithms generally produce better results with more data available from which to learn. Therefore, prediction accuracy can be further improved by incorporating other data. In our opinion, there are two promising kinds of data. One kind of data is actor's expertise information and their social relationship [18,24]. The approach presented in Ref. [5] is an example of using such kind of information. The other kind of data is workflow application data. John and Langley work [21] can be viewed as an example.

For the first kind of data, its applicability depends on how well the expertise of actors and their relationship information can be expressed by the system, which inevitably leads to the discussion of the sophisticated capability model for people and organizations. Although there are many kinds of models that are presented in the literature, most of them are actually difficult to implement for current workflow systems. Besides, the cost of practicing such kinds of models is rather expensive, hence the availability of this part of information is a big issue.

For the second kind of data, the workflow application data are very promising to be incorporated into learning, at least, its availability is much better than the expertise information. The obstacle faced is that data are usually application context dependent, which means that different enterprise's data contain different information. Therefore, special approaches are needed to extract useful feature information, for example, using process data warehouse [25]. Nevertheless, the feedback from enterprise users has also revealed the fact that people often

Table 6
The best and worst predicted activities in three enterprises

Enterprise	Number of actors	Training samples	Number of features	Prediction accuracy (%)		
				C4.5	Naive Bayes	SVM
A						
Worst	29	49	1	64.50	32.50	64.50
Best	2	33	14	93.33	87.50	100.00
B						
Worst	30	148	2	21.48	23.52	21.52
Best	6	25	1	100.00	100.00	100.00
C						
Worst	3	11	12	70.00	40.00	60.00
Best	8	298	1	100.00	100.00	100.00

make their decisions based on the documents that are processed by the workflow. This fact motivates us to carry out further investigation.

In the real situation, the workflow event log is not available all at a time, however, the approach we present in this paper trains the classifier using a batched set of data. Alternatively an incremental algorithm could be used whereby instances are provided one at a time to the classifier and the classifier updates itself accordingly [26]. This incremental approach will be more suitable than the batched approach especially for those enterprises whose workflow system is intensively used.

Moreover, the approach we have presented in this paper only recommends potential actors one at a time. As a matter of fact, groups of actors often work on similar kinds of activities. Therefore, it might be more helpful to recommend a small list of potential actors rather than just one. However, determining the best group of actors is not as easy as it seems to be. Further experiments and development of the learning approach are needed.

6. Related work

Our work is related to workflow resource management. Although resource management is very important in workflow applications, most work in the field of workflow focuses on the control and data perspectives. Only a relatively small body of research into the resource aspect is presented [27,28,3].

To the best of our knowledge, the research that is most similar to ours is presented in Refs. [5,6]. In Ref. [5], Ly et al. show that the problem of deriving staff assignment rules using information from event log data and organizational information as input can be interpreted as an inductive learning problem. Therefore, machine learning techniques can be adapted in order to solve the problem. In particular, they have used decision tree learning on simulated datasets to derive meaningful staff assignment rules. Later on, in Ref. [6], Rinderle and van der Aalst extended the work of staff assignment rule mining to provide a lifecycle support for staff

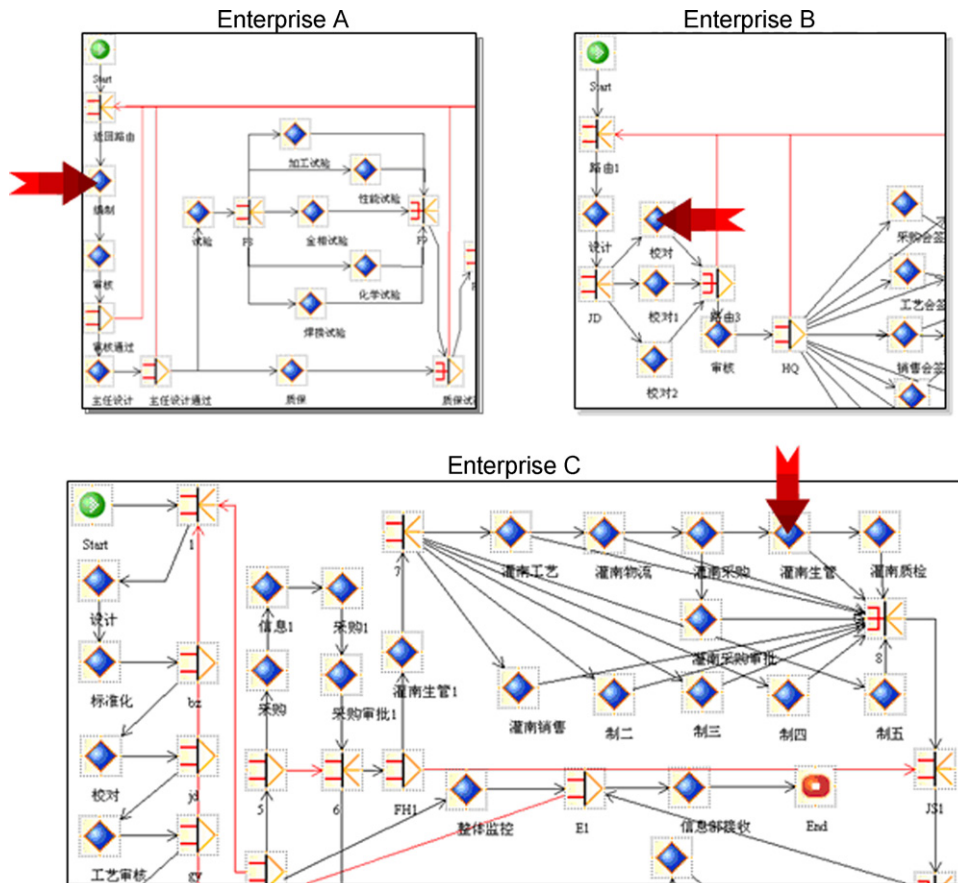


Fig. 9. Some workflow model fragments of worst predicted activities in three enterprises.

assignment rules and they implemented the approach in the context of ProM framework.

Our work is different from theirs in that our work focuses on automating the staff assignment performed at workflow run-time stages. In addition, we evaluate our approach on real datasets using three different learning algorithms, which further demonstrates the applicability of using machine learning in workflow staff assignment.

In Refs. [29,30], Shan et al. present a resource management facility, which is implemented in the HP workflow system: Changengine. Their approach consists of an independent resource manager that integrates existing local and site resource management systems under a common schema. The proposed system implements an SQL-like language for the querying and assignment of resources.

In Ref. [31], as part of the XRL/Woflan project which aims at the development of an exchangeable workflow modeling language, van der Aalst et al. propose a set of UML diagrams to specify workflow resource models. This UML diagram can be expressed using XML tags to facilitate information exchange.

In Ref. [4], zur Muehlen gives a comprehensive overview on the organizational aspect about workflow technology in the context of the workflow life cycle. He outlines variations in the assignment and synchronization policies for the run-time allocation of activities to performers, and presents different strategies for the actual activity assignment at run-time. He presents strategies for the use of workflow event log data to improve resource utilization and development of new process strategies. He also mentions the idea of knowledge-based resource allocation algorithms, but he does not provide any detailed information about the implementation.

Russell et al.'s work of workflow resource patterns can be viewed as a more comprehensive study about resource management of contemporary workflow products [2,3]. They investigate the resource management mechanism of five workflow products: Staffware, WebSphere MQ Workflow, FLOWer, COSA and iPlanet, and propose 43 types of workflow resource patterns. These patterns can be classified into six categories: creation pattern, push pattern, pull pattern, detour pattern, auto-start pattern and visibility pattern. All these patterns can be viewed as a guideline for the workflow management system construction. Like workflow control pattern and data pattern, their work is also a part of workflow pattern research [32,33].

In spite of the research about resource allocation in the field of workflow, the idea of automatic resource allocation can also be found in other areas. In Ref. [21], John and Langley propose a supervised machine learning approach for automating bug fixing assignment in open source development projects. They extract the feature information and related manual assignment records from historical bug reports, and apply three machine learning algorithms to derive the bug assignment rules. In their experiments, they validate the applicability of their approach in the Firefox and Eclipse projects and reach precision levels of 64% and 57%, respectively.

Another research about automatic resource allocation is proposed by Reis et al. [34]. They focus on process-centered

software engineering environments (PSEEs). In their paper, they describe a mechanism to automatically assist software process instantiation through user-defined policies. When the process is instantiated, the resource is automatically suggested as an ordered list. This mechanism utilizes a meta-model to describe the attributes about resource and software process. The policy is then defined using a language based on this meta-model. After the policy has been defined, the planner component interprets the policy and suggests an ordered list of resource for each activity. Although they describe the architecture and provide a prototype example, they do not describe how to generate the policies. According to their description, this work mainly depends on manual work.

The problem of finding appropriate person is also addressed by the researchers in the security field, like the RBAC model [35], but their objective is focused on the authorization and the security related issues [3].

7. Summary

In this paper, we have discussed an approach to semi-automating the run-time staff assignment in workflow management systems. Our approach uses supervised machine learning algorithms that are applied to the workflow event log. In the experiments on three enterprises' datasets, our approach has achieved an overall prediction accuracy of well over 75%, which clearly demonstrates the feasibility of our approach. In addition to presenting our results, we have analyzed the performance between different learning algorithms and discussed the limitation of our approach and possible improvement.

We believe our approach can be used at workflow run-time instantiation and execution stages to improve current state of workflow staff assignment. Our future work includes further investigation on additional sources of information, exploration on the experiments results and empirical study of the usage of our approach in enterprises.

Acknowledgements

We are grateful to Tsinghua InfoTech Company, Xiamen King Long United Automotive Co. Ltd., Hebei ZhongXing Automobile Co. Ltd. and Datong Electronic Locomotive Co. Ltd. for providing us the workflow history data of their TiPLM system. We would also like to thank the reviewer's helpful comments. This paper is a significantly extended version of Ref. [16].

This work is partly supported by the National Basic Research Program of China (No. 2007CB310802), the Project of NSFC (No. 60373011 and No. 60473077), the Project of 973 Program of China (No.2002CB312006) and by the Program for New Century Excellent Talents in University, Ministry of Education, China.

References

- [1] F. Leymann, D. Roller, *Production Workflow: Concepts and Techniques*, 1st ed., Prentice Hall PTR, 1999.

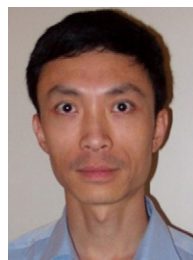
- [2] N. Russell, A.H.M. ter Hofstede, D. Edmond, W.M.P. van der Aalst, Workflow Resource Patterns, Eindhoven University of Technology, Eindhoven, 2005.
- [3] N. Russell, W.M.P. van der Aalst, A.H.M. ter Hofstede, D. Edmond, Workflow resource patterns: identification, representation and tool support, Lecture Notes in Computer Science (2005) 216–232.
- [4] M. zur Muehlen, Organizational management in workflow applications—issues and perspectives, Information Technology Management 5 (2004) 271–291.
- [5] L.T. Ly, S. Rinderle, P. Dadam, M. Reichert, Mining staff assignment rules from event-based data, Lecture Notes in Computer Science 3812 (2006) 177–190.
- [6] S. Rinderle, W.M.P. van der Aalst, Life-Cycle Support for Staff Assignment Rules in Process-Aware Information Systems, Department of Technology Management, Eindhoven University of Technology, Eindhoven, 2007.
- [7] Introduction to Infotech Product Lifecycle Management Solution (in Chinese), <http://www.thit.com.cn/TiPLM/TiPLM.htm>.
- [8] CIMdata, About Product Lifecycle Management, <http://www.cimdata.com/PLM/aboutPLM.html>.
- [9] Xiamen King Long United Automotive Co. Ltd. (Home Page), <http://www.king-long.com.cn/>.
- [10] Hebei Zhongxing Automobile Co. Ltd. (Home Page), <http://www.zxauto.com.cn/>.
- [11] Datong Electronic Locomotive Co. Ltd. (Home Page), <http://www.dtlco.com/>.
- [12] J. Han, M. Kamber, Data Mining: Concepts and Techniques, 1st ed., Morgan Kaufmann, San Francisco, 2001.
- [13] D. Grigori, F. Casati, U. Dayal, M.-C. Shan, Improving business process quality through exception understanding, in: Prediction, and Prevention in Proceedings of the 27th VLDB Conference, Roma, Italy, (2001), pp. 159–168.
- [14] D. Grigori, F. Casati, M. Castellanos, U. Dayal, M. Sayal, M.-C. Shan, Business process intelligence, Computers in Industry 53 (2004) 321–343.
- [15] M. Bennour, D. Crestani, O. Crespo, F. Prunet, Computer-aided decision for human task allocation with mono- and multi-performance evaluation, International Journal of Production Research 43 (2005) 4559–4588.
- [16] Y. Liu, J. Wang, J. Sun, A machine learning approach to semi-automating workflow staff assignment, in: The 22nd Annual ACM Symposium on Applied Computing (SAC2007), Seoul, Korea, (2007), pp. 340–346.
- [17] R.K. Ahuja, T.L. Magnanti, J.B. Orlin, Network Flows: Theory, Algorithms, and Applications, Prentice Hall, United States, 1993.
- [18] W.M.P. van der Aalst, M. Song, Mining social networks: uncovering interaction patterns in business processes, in: Business Process Management: Second International Conference, BPM 2004, Potsdam, Germany, (2004), pp. 244–260.
- [19] W.M. Coalition. The Workflow Management Coalition Specification: Audit Data Specification, <http://www.wfmc.org/>. 1996.
- [20] R. Quinlan, C4.5: Programs for Machine Learning, 1st ed., Morgan Kaufmann Publishers, San Mateo, CA, 1993.
- [21] G.H. John, P. Langley, Estimating continuous distributions in bayesian classifiers, in: Eleventh Conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann, San Mateo, (1995), pp. 338–345.
- [22] J. Platt, Fast training of support vector machines using sequential minimal optimization, in: B. Schoelkopf, C. Burges, A. Smola (Eds.), Advances in Kernel Methods—Support Vector Learning, MIT Press, 1998.
- [23] I.H. Witten, E. Frank, Data Mining: Practical Machine Learning Tools and Techniques, 2nd ed., Morgan Kaufmann, San Francisco, 2005.
- [24] W.M.P. van der Aalst, H.A. Reijers, M. Song, Discovering social networks from event logs, Computer Supported Cooperative Work (CSCW) 14 (2005) 549–593.
- [25] F. Casati, M. Castellanos, U. Dayal, N. Salazar, A generic solution for warehousing business process data, in: 33rd International Conference on Very Large Data Bases, Vienna, Austria, (2007), pp. 1128–1137.
- [26] R. Segal, J. Kephart, Incremental learning in swiftfile, in: The Seventh International Conference on Machine Learning, 2000, 863–870.
- [27] A. Kumar, W.M.P. van der Aalst, H.M.W. Verbeek, Dynamic work distribution in workflow management systems: how to balance quality and performance? Journal of Management Information Systems 18 (2002) 157–193.
- [28] M. Petic, W.M.P. van der Aalst, Modeling work distribution mechanisms using colored Petri nets, in: Proceedings of the Sixth Workshop on the Practical Use of Coloured Petri Nets and CPN Tools (CPN 2005), Aarhus, Denmark, (2005), pp. 157–176.
- [29] W. Du, M.-C. Shan, Enterprise workflow resource management, HP Labs Technical Reports, 1999, pp. 108–115.
- [30] Y.-N. Huang, M.-C. Shan, Policies in a resource Manager of workflow systems: modeling, enforcement and management, HP Labs Technical Reports, 1999, p. 103.
- [31] W.M.P. van der Aalst, A. Kumar, H.M.W. Verbeek, Organizational modeling in Uml and Xml in the context of workflow systems, in: Symposium on Applied Computing (SAC) 2003, Melbourne, FL, USA, (2003), pp. 603–608.
- [32] W.M.P. van der Aalst, A.H.M. ter Hofstede, B. Kiepuszewski, A.P. Barros, Workflow Patterns, Distributed Parallel Database 14 (2003) 5–51.
- [33] N. Russell, A.H.M. ter Hofstede, D. Edmond, W.M.P. van der Aalst, Workflow Data Patterns, Queensland University of Technology, 2004.
- [34] C.A.L. Reis, R.Q. Reis, H. Schlebbe, D.J. Nunes, A policy-based resource instantiation mechanism to automate software process management, in: Proceedings of the 14th International Conference on Software Engineering and Knowledge Engineering, Ischia, Italy, (2002), pp. 795–802.
- [35] D.F. Ferraiolo, R. Sandhu, S. Gavrila, D.R. Kuhn, R. Chandramouli. Proposed nist standard for role-based access control. ACM Trans. Inform. Syst. Secur.; 4:224–274.



Yingbo Liu is a PhD candidate in the Department of Computer Science and Technology at Tsinghua University. He received his BS from National University of Defense Technology, China. His research interests are software engineering and distribute computing. He has many practices at the design and development of workflow management systems, product lifecycle management and CIMS engineering.



Jianmin Wang is director of Research Institute of Information System and Engineering, and Deputy Dean of School of Software, Tsinghua University. He is a prof. of computer science and a senior expert for China's 863 Hi-Tech Research and Development Program. His current research interests include organizational information systems, database and workflow technology, software measurement and testing, data grid and data mining. And he has conducted numerous China's national telecommunication projects. Before coming to join Tsinghua faculty, he worked for Tsinghua Tong Fang Software Co. Ltd. as a senior system architect. Prof. Wang received his PhD in computer software from Tsinghua University and BS in computer science from Peking University.



Yun Yang received his BS and MEng from Anhui University and the University of Science and Technology of China, Hefei, China, in 1984 and 1987, respectively, and his PhD degree from the University of Queensland, Brisbane, Australia, in 1992, all in computer science. He is currently a full prof. and leader of workflow technology research program in the faculty of information and communication technologies at Swinburne University of Technology, Melbourne, Australia.

Prior to that he held positions at Deakin University and CRC for distributed systems technology, and Beijing University of Aeronautics and Astronautics. He has published more than 160 papers on journals and refereed conferences. His research interests include software technologies, p2p and grid workflow systems, service-oriented computing, internet computing applications,

CSCW, and e-business and e-science processes. He is a member of IEEE and IEEE Computer Society.



Jianguang Sun is a member of the Chinese Academy of Engineering and Dean of School of Software, assumed the post of director of Tsinghua National Lab for Information Science and Technology (TNLIST). He is also president of Executive Council of China Engineering Graphics Society and vice president of National Natural Science Foundation of China. Prof. Sun was born in ZhenJiang of Jiangsu province in 1946 and graduated from Department of Automation, Tsinghua University in 1970. Prof. Sun has engaged in

the research and teaching of computer graphics, computer aided design and computer aided management since 1975. He has developed six software products including computer aided design and drawing, 3D geometry modeling, integrated CAD/CAM supporting software, engineering drawing management, enterprise resource planning and product data management system. He has also made innovation in product data management system based on corba/web, geometry modeling with wireframe, surface, solid and feather, environment for multi-resource, multi-event and multi-process working cooperatively. The project cooperative geometry design which he co-applied was supported by National Natural Science Foundation of America and National Natural Science Foundation of France. Prof. Sun has won the Science and Technology Development Awards fifteen times awarded by the Nation and the Ministry. He has published many academic papers. Prof. Sun also published four books as the first author.