A Role-based Adjacent Workflow Recommendation Technique

Guangzhi Chen
School of Information Science and Technology
Sun Yat-Sen University
Guangzhou, P.R.China
chgzhi@mail2.sysu.edu.cn

Lei Li
School of Information Science and Technology
Sun Yat-Sen University
Guangzhou, P.R.China
lnslilei@mail.sysu.edu.cn

***Abstract*—Recently recommendation techniques and systems have gained more and more attention and are researched in a variety of fields. In this paper we put the recommendation technique into the problem of workflow or Web browsing so that can facilitate the workflow participants to find their next most desired activities or the people in the Business Internet to find their next wanted clicks. The distinguished feature of our automated workflow recommendation technique is that the concept of role-based adjacent workflow instance is incorporated. Our proposed algorithm first calculates the multi-set of role-based adjacent workflow instances of an incomplete workflow instance, and then utilizes this multi-set to predict the next desired activity for that incomplete workflow instance. The experimental results confirm the promising effectiveness of our role-based adjacent workflow recommendation algorithm.**

Keywords—activity; role-based adjacency; workflow instance; workflow recommendation

I. INTRODUCTION

In recent years, recommendation techniques and systems have gained more and more attention, and are researched and applied in a variety of fields [1-5]. Upon that background it has a practical benefit to put the recommendation techniques into workflow execution. A business website, e.g. Amazon, its back-end database might accumulate a wealth of users' click information, when a new user comes to this website to choose some product to buy, it is better to utilize recommendation technique to help that user quickly locate his desired product by the website's history user click information. A whole browsing process for a user can be thought of as a workflow instance, and every click operation for that user can be thought of as an activity. In the Service Oriented Computing domain, the nodes (Web services) that are appropriate to the task that are to be fulfilled are first discovered and the best Web service(s) is then identified for execution in the workflow through some service optimization process [3]. In a sense, this resembles to a workflow recommendation problem where the service discovery and optimization help recommend to users the best service that needs to be execute in each step [4]. Whereas discovering desired services manually can be time consuming, tedious, and costly.

Another motivation for our paper is that, in government approval domain, although some specific project's business process is predefined, several execution instances of that business process will violate the predefined business rules. In this case, using history execution instances of a government approval project to recommend the next desired activity for an applicant approaches the real intention of the business more than that of using only the predefined business rules.

From the aforementioned concrete problems we abstract their common problem formulation, and present a novel workflow recommendation algorithm, called role-based adjacent workflow recommendation algorithm (RoleB). Algorithm RoleB recommends the next desired activity for an incomplete workflow instance by utilizing an execution instance repository. The core of algorithm RoleB is to search the role-based adjacencies of an incomplete workflow instance, further speaking, RoleB uses these adjacencies to calculate the next desired activity for that instance. Our algorithm RoleB does not confine to workflow recommendation, it can also recommend the next desired click for the users that surf on a business website.

A synthetic dataset used as the execution instance repository is generated to verify whether or not the algorithm RoleB is effective. For better comparison, we also design a random workflow recommendation algorithm (RanB). Our experimental results show that RoleB outperforms algorithm RanB. The contributions of this paper can be summarized as follows: present the concept of role-base adjacent workflow instance; design a novel algorithm RoleB; compare and contrast the recommendation accuracy of algorithm RoleB and RanB, to validate the effectiveness of algorithm RoleB.

The rest of this paper is organized as follows. Section II presents the related concepts and gives problem formulation. In Section III, algorithm RoleB is proposed and discussed. The experimental results are reported in Section IV. The final section concludes the whole paper and presents some future research directions.

II. RELATED CONCEPTS AND PROBLEM FORMULATION

A. Related concepts and background knoweledge

First, in this section we will give some related concepts and background knowledge.

Definition 1 A *workflow model* is a 6-tuple $\Omega \equiv (T, R, t_s, t_e, L, l)$, where T is a set of tasks; $R \subseteq (T \times T)$ is a set of directed edges; L is a set of labels which give meaning for each task; t_s is a start task, its in-degree is 0; t_e is a terminated task, its out-degree is 0; $l : T \rightarrow L$ is a function, it assigns each task in T a label in L. A task is called a *conditional branch*, if its out-degree is greater than 1; a task is called a *loop branch*, if one of directed edges which originated from it directs toward a prior task.

Example 1 Fig. 1 is a simple example of a workflow model, v_1, ..., v_7 is the tasks' id, and the words next to each task node is the label of that task. Task v_2 is a conditional branch, v_6 is loop branch. Note that our definition of workflow model is a simplified version of the definition of Workflow nets in literature [6]. The omitting contents include conditions for conditional branch and loop branch. For example, the condition for transferring task v_6 to task v_2 maybe that a new problem is found after post auditing or the application is incomplete. Since these kinds of contents are irrelevant to our research problem, omitting them makes us focus on the main problem. We also do not incorporate the role of workflow participant in the definition which will be discussed in the following section.

The loop branch v_6 in Example 1 is different from the loop in programming languages. Whereas iteration number in programming languages can be as large as thousands of hundreds, the iteration number of loop branch in a workflow model can only be a few. The reason for presenting the definition of the workflow model is that a workflow instance repository can only be generated according to a predefined workflow model. Our algorithm is only related to the workflow instance repository, the workflow model behind the workflow instance repository is irrelevant.

A workflow participant with a role executes a business process step-by-step according to the predefined workflow model until the terminated task is reached; this execution trace is called a workflow instance of that workflow model. Therefore, many different workflow instances can be generated from a common workflow model. A task in a workflow instance is called an activity in this paper.

Definition 2 Let Ω be a workflow model, a *workflow instance* ω generated from Ω is a 2-tuple (r, A), where r is the role of workflow participant, $A \equiv \langle a_1, a_2, ..., a_n \rangle$ is a sequence of activities that is executed by a workflow participant with role r, and activities a_1 and a_n conform to the start task t_s and terminated task t_e of Ω respectively.

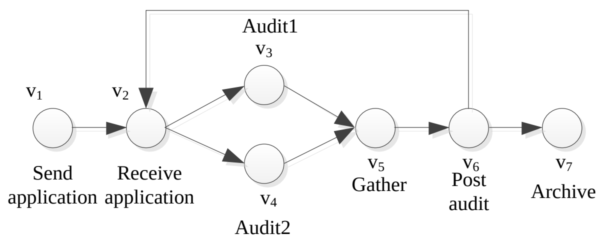

Fig. 1. An example of workflow model.

Example 2 Assume that $R = \{plain, VIP\}$ is a set of roles for all possible workflow participants, then two workflow instances generated from workflow model in Example 1 may be:

$$\omega_1: (plain, \langle v_1, v_2, v_3, v_5, v_6, v_2, v_4, v_5, v_6, v_7 \rangle) \qquad (1)$$

$$\omega_2: (VIP, \langle v_1, v_2, v_3, v_5, v_6, v_7 \rangle) \qquad (2)$$

Next we will discuss our key concept of role-based adjacency, it plays an important role in algorithm RoleB.

Definition 3 Let $\omega_1 = (r_1, A_1)$ and $\omega_2 = (r_2, A_2)$ be two workflow instances generated from a common workflow model Ω, if ω_1 and ω_2 satisfy the following two conditions:

1. r_1 and r_2 are the same role;
2. $|A_1^s \cap A_2^s| \geq \delta$, where A_1^s is a set whose elements come from activity sequence A_1, so A_2^s; $|A^s|$ represents the number of elements in set A^s; δ is a predefined threshold;

then ω_1 and ω_2 are called mutually *role-based adjacent* workflow instances, denoted by $\omega_1 \ {}_{adj}\!\sim \ \omega_2$. We also say that ω_1 and ω_2 are role-based adjacent, and ω_1 and ω_2 are called role-based adjacency of ω_2 and ω_1 respectively.

According to Definition 3, set A^s of ω_1 in Example 2 is $\{v_1, v_2, v_3, v_4, v_5, v_6, v_7\}$, whereas set A^s of ω_2 in Example 2 is $\{v_1, v_2, v_3, v_5, v_6, v_7\}$. From Definition 3 we also know that the role-base adjacency of a workflow instance is determined by threshold δ, different value δ will produce different role-based adjacency of a workflow instance.

Example 3 For workflow model in Example 1, if we make $\delta = 3$, then the workflow instance ω_1 in (1) and workflow instance

$$\omega_3: (plain, \langle v_1, v_2, v_3, v_5, v_6, v_7 \rangle) \qquad (3)$$

are role-based adjacent; but workflow instance ω_2 in (2) and ω_3 are not role-based adjacent.

Definition 4 If the workflow participant does not reach the terminated task of a workflow model Ω, just in some intermediate task, this kinds of execution trace is call an *incomplete workflow instance* of Ω, denoted by $\hat{\omega} = (r, \hat{A})$, i.e., it satisfies the following two conditions:

1. role r is known;
2. $\hat{A} = \langle a_1, a_2, ..., a_j \rangle$, where activity a_j is not the terminated task of the workflow model Ω.

B. Problem formulation

Assume that we have a workflow instance repository $I_\omega = \{\omega_1, \omega_2, ..., \omega_k\}$ of a workflow model Ω, where k is natural number greater than 1, and an incomplete workflow instance $\hat{\omega} = (r, \hat{A})$ of Ω, where $\hat{A} = \langle a_1, a_2, ..., a_j \rangle$. Our problem is that how to recommend the next desired activity a_{j+1} for $\hat{\omega}$ by utilizing the information in I_ω.

III. Workflow Recommendation

Before presenting our algorithm RoleB, we will give the following two definitions so that RoleB can be formulated succinctly.

Definition 5 Let $I_\omega = \{\omega_1, \omega_2, ..., \omega_k\}$ be a workflow instance repository of some workflow model Ω, and $\hat{\omega}$ be an incomplete workflow instance of Ω, a subset of I_ω is called a *candidate workflow recommendation set* of $\hat{\omega}$, if every element in this subset is a role-based adjacency of $\hat{\omega}$, denoted by $C_{\hat{\omega}}$, i.e., $C_{\hat{\omega}} = \{ \omega_{j1}, \omega_{j2}, ..., \omega_{js} \} \subseteq I_\omega$, where s is a natural number less than k, $1 \le j_i \le k$ (i = 1, 2, ..., s) is natural number, for any $\omega_{ji} \in C_{\hat{\omega}}$, $\hat{\omega}_{adj} \sim \omega_{ji}$.

Definition 6 Let $\hat{\omega} = (r, <a_1, a_2, ..., a_j>)$ be an incomplete workflow instance of some workflow model Ω, $C_{\hat{\omega}} = \{ \omega_1, \omega_2, ..., \omega_s \}$ be a candidate workflow recommendation set of $\hat{\omega}$, where $\omega_i = (r, <a_{i,1}, a_{i,2}, ..., a_{i,j+1}, ...>)$, i = 1, 2, ..., s, then the multi-set whose elements are $a_{i,j+1}$ (i = 1, 2, ..., s) is called *forward activity recommendation set* of $\hat{\omega}$, denoted by $F_{a,\hat{\omega}}$.

A. Size-increasing δ function

For an incomplete workflow instance $\hat{\omega} = (r, \hat{A})$, we call the value $|\hat{A}^s|$ the size of $\hat{\omega}$. Obviously, we should increase the threshold δ with the increasing size of $\hat{\omega}$ during calculating its candidate workflow recommendation set $C_{\hat{\omega}}$. Otherwise, the candidate workflow recommendation set $C_{\hat{\omega}}$ calculated will be the same for different $\hat{\omega}$ with different size which is greater than δ. In this case, we cannot get the real role-based adjacency of $\hat{\omega}$.

To circumvent the above problem, our solution is that to make the threshold δ increasing with $|\hat{A}^s|$ and assure $|\hat{A}^s| \le \delta$. Thus, the concept of size-increasing δ function will be introduced.

Definition 7 For two incomplete workflow instances $\hat{\omega}_1 = (r, \hat{A}_1)$ and $\hat{\omega}_2 = (r, \hat{A}_2)$ with the same role r, $|\hat{A}_1^s|$ and $|\hat{A}_2^s|$ are their sizes respectively, a monotonically increasing function $\varphi(x)$ is called a *size-increasing δ function*, if it satisfies the following two conditions: for $|\hat{A}_1^s| < |\hat{A}_2^s|$, $\varphi(|\hat{A}_1^s|) < \varphi(|\hat{A}_2^s|)$; $x \le \varphi(x)$.

The size-increasing δ function $\varphi(x)$ is used in algorithm RoleB, so that the more similar role-based adjacencies of $\hat{\omega}$ can be calculated. In this paper we will use $\varphi(x) = x$ as our size-increasing δ function, other kinds of $\varphi(x)$ with different format can be used in algorithm RoleB as soon as they conform to Definition 7.

B. Role-based adjacent workflow recommendation algorithm

Role-based adjacent workflow recommendation algorithm, which is abbreviated for RoleB, is referred to Algorithm 1. This algorithm first calculates the size of the incomplete workflow instance $\hat{\omega}$ and function $\varphi(|\hat{A}^s|)$ to get the threshold δ for this specific $\hat{\omega}$; after the threshold δ is known, candidate workflow recommendation set $C_{\hat{\omega}}$ and forward activity recommendation set $F_{a,\hat{\omega}}$ are calculated; finally, a randomly selected activity from multi-set $F_{a,\hat{\omega}}$ is returned.

The bottleneck of algorithm RoleB is the step 3, for every workflow instance ω_i in I_ω, we must calculate the intersection of A_i^s and $\hat{A}^s$, and then compare the value $|A_i^s \cap \hat{A}^s|$ with the threshold δ to determine whether or not ω_i is a role-based adjacency of $\hat{\omega}$. Let $I_\omega = \{\omega_1, \omega_2, ..., \omega_k\}$, and $|A^s| = \max(|A_1^s|, |A_2^s|, ..., |A_k^s|)$, since $\hat{\omega}$ is an incomplete workflow instance, it follows that $|\hat{A}^s| < |A^s|$, so in the worst cast of all the workflow instance in I_ω having the same role with $\hat{\omega}$, the time complexity of algorithm RoleB is $O(k|A^s|)$.

Algorithm 1: RoleBasedAdjacencyRecommendation

Input: a workflow instance repository I_ω of some workflow model Ω; an incomplete workflow instance $\hat{\omega}$; size-increasing δ function $\varphi(x)$.

Output: the next desired activity a^* for $\hat{\omega}$.

1 calculate $|\hat{A}^s|$ of $\hat{\omega}$;

2 calculate $\varphi(|\hat{A}^s|)$, $\delta \leftarrow \varphi(|\hat{A}^s|)$;

3 traverse I_ω to calculate $C_{\hat{\omega}}$ with δ;

4 calculate $F_{a,\hat{\omega}}$ through $C_{\hat{\omega}}$;

5 randomly select an activity a^* from $F_{a,\hat{\omega}}$;

6 return a^*;

IV. Experimental Results and Analysis

For better evaluating the effectiveness of RoleB, we also implement the random workflow recommendation algorithm (RanB). Its main idea is as follows: 1) randomly select a workflow instance ω_i from I_ω; 2) if the length of activity sequence of ω_i is greater than the length l of activity sequence of $\hat{\omega}$, then put the activity a_{l+1} of ω_i into multi-set $F_{a,\hat{\omega}}$; 3) repeat the step 1) and 2) k times (k is the number of workflow instances in I_ω); 4) randomly select an activity a^* from $F_{a,\hat{\omega}}$ as the next recommendation activity for $\hat{\omega}$.

The program for algorithm RoleB and RanB is developed in C++ and all the experiments are conducted in Windows 7 2G system with a main memory 2G.

A. Experimental dataset

The workflow instance repository I_ω that we will use in the experiments is generated synthetically. The process for generating I_ω is as follows: first we manually design a workflow model, its representation is similar to literature [7], it contains 18 tasks, one three-conditional branch, one two-conditional branch, one loop branch; then generate k activity sequences (k is the number of workflow instances in I_ω); finally, assign role to every activity sequence for the above generated k activity sequences. For two activity sequences A_i and A_j, if the following condition is satisfied:

$$(||A_i^s| - |A_i^s \cap A_j^s|| \le 2) \text{ or } (||A_j^s| - |A_i^s \cap A_j^s|| \le 2),$$

then A_i and A_j will be assigned to a same role. The names of roles used during generating process are r1, r2, ..., etc.

In our experimental setting, k = 1000, i.e., 1000 workflow instances are generated, and finally four roles r1, r2, r3, and r4 are assigned to these instances. The condition for assigning role to activity sequence assures that workflow instance repository generated as being close to the real-life application scenarios as possible. In real-life application, two workflow instances with the same role will have many common activities.

B. Experimental method and results

We select four workflow instances from I_ω with four different roles from which incomplete workflow instances are constructed. Denote these four workflow instances by ω_1, ω_2, ω_3, and ω_4, incomplete workflow instances with different length (2, 3, 4, 6, 7, 8, 11, 13, 15, and 17) of activity sequence are constructed from them, we use symbol $\omega_{i,j}^{\wedge}$ to represent an incomplete workflow instance that is constructed from ω_i and with length j. Since the $(j+1)^{th}$ activity of ω_i represents the next real activity of $\omega_{i,j}^{\wedge}$, we can compute the recommendation accuracy of algorithms RoleB and RanB. If $|F^{*}_{a,\ \omega i,j^\wedge}|$ represents the number of next real activity of $\omega_{i,j}^{\wedge}$ in multi-set $F_{a,\ \omega i,j^\wedge}$, then the recommendation accuracy of an algorithm for $\omega_{i,j}^{\wedge}$ can be calculated by:

$$\text{Accuracy}(\omega_{i,j}^{\wedge}) = |F^{*}_{a,\ \omega i,j^\wedge}| \ / \ |F_{a,\ \omega i,j^\wedge}|. \qquad (4)$$

Fig. 2 and Fig. 3 give the comparison of recommendation accuracies for incomplete workflow instances $\omega_{1,j}^{\wedge}$ and $\omega_{4,j}^{\wedge}$ respectively. From Fig. 2 and Fig. 3 we can see that our algorithm RoleB outperforms algorithm RanB. For plain activity, the recommendation accuracies of RoleB are all 1, but for conditional branch activity or loop branch activity, recommendation accuracies of RoleB are decreased significantly; whereas recommendation accuracies of RanB are insensitive to the change of types of activities.

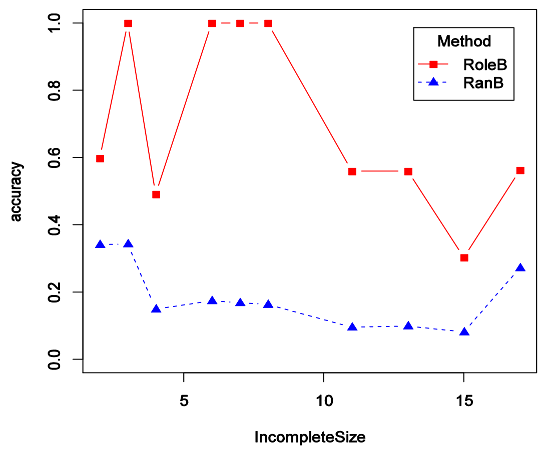

Fig. 2. Comparison of recommendation accuracies for incomplete workflow instance with role r1.

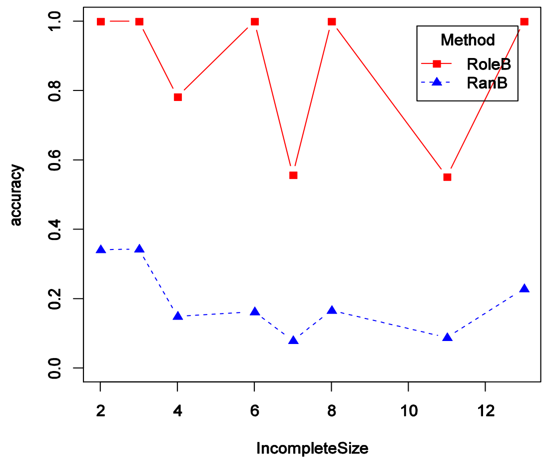

Fig. 3. Comparison of recommendation accuracies for incomplete workflow instance with role r4.

V. CONCLUSION

In this paper, we propose a novel workflow recommendation algorithm, called RoleB, that utilizes role-based adjacencies of an incomplete workflow instance to recommend the next most desired activity for that incomplete instance. Our workflow recommendation technique can be applied in government approval, business website browsing etc. The experimental results show that our technique is effective. In the future, we will design some techniques for improving the recommendation accuracy of conditional branch activities and loop branch activities.

REFERENCES

[1] K. Mao, L. Shou, J. Fan, G. Chen, and M.S. Kankanhalli, “Competence-based song recommendation: matching songs to one’s singing skill,” IEEE. Trans. Multimedia, vol. 17, pp. 396-408, March 2015.

[2] B. Liu, H. Xiong, S. Papadimitriou, Y. Fu, and Z. Yao, “A general geographical probabilistic faccctor model for point of interest recommendation,” IEEE. Trans. Knowledge and Data Engineering, vol. 27, pp. 1167-1179, May 2015.

[3] U.S. Manifrao and T.V. Prabhakar, “Dynamic selection of Web Services with recommendation system,” in International Conference on Next Generation, August 2005.

[4] J. Zhang, Q. Liu, and K. Xu, “Flowrecommender: a workflow recommendation technique for process provenance,” in Conferences in Research and Practice in Information Technology Series, pp. 55-61, 2009.

[5] A. Koschmider, T. Hornung, and A. Oberweis, “Recommendation-based editor for business process modeling,” Data and Knowledge Engineering, vol. 70, pp. 483-503, 2011.

[6] W.M.P. van der Aalst, K.M. van Hee, A.H.M. ter Hofstede, N. Sidorova, H.M.W. Verbeek, M. Voorhoeve, and M.T. Wynn, “Soundness of workflow nets: classification, decidability, and analysis,” Formal Aspects of Computing, vol. 23, pp. 333-363, 2011.

[7] W. Han, J. Lee, M. Pham, and J. Xu Yu, “iGraph: a framework for comparisons of disk-based graph indexing techniques,” PVLDB, vol. 3, 2010.