An Extended CompDepend-Algorithm to Discovery Duplicate Tasks in Workflow Process Mining System*

Wang XiaoHui, Zhao WenQing
School of Computer Science & Technology
North China Electric Power University
Baoding, China
maconi@126.com

Guo FengJuan
School of Science & Technology
North China Electric Power University
Baoding, China
gfj0917@126.com

Abstract— **The workflow mining technology had become a hot research field among computer application. In current mining algorithm, duplicated tasks are hard to be modeled. This paper proposes an extended CompDepend-Algorithm able to discovery duplicate tasks. Firstly, a new data structure based on sequence set is designed to describe duplicate tasks. Secondly, rules to discover duplicated tasks based on dependency relationship are proposed. Then the algorithm of workflow mining based these rules is given. At last, Experimental results prove that the algorithm is effective.**

Keywords-component;workflow mining; duplicate tasks; extended directed graph; sequence set

I. INTRODUCTION

With widely application of workflow technology, many manufactories has a lot of workflow event log data. Due to rapid change of business circumstances, business processes are characterized by uncertainty and variability. And the manual modeled processed maybe obsolete quickly. The purpose of workflow mining is to build proper model based on event-based logs and to support the analysis and design of workflow models[1]. Therefore workflow mining technology has becoming the hotspot research domain.

Cook [3] firstly discovered models of software processes in software engineering from event-based data, and presented three software processes mining methods: one using neural networks, one using a pure algorithmic approach, and one Markovian approach. Agrawal [4] firstly introduced the concept of process mining to workflow system and applied an algorithm to process logs obtained from an IBM's workflow management platform – MQSeries[5]. After that research on workflow mining deeply developed.

Reference [7] proposed 3 mining algorithms based on state flow, but these algorithms can only describe sequence processes models. Reference [8] design a mining algorithm based on directed diagram, which data structure is activities set. This algorithm can find sequence, direct loop and causality process model quickly and efficiently. But it can not deal with duplicate tasks. This paper proposed an extended algorithm to solve this problem. A data structure based on sequence set is proposed to describe duplicate tasks, and the Extended-CompDepend-Algorithm is proposed.

*This work is supported by the Teacher's Research Foundation of North China Electric Power University (200811014).

II. CONCEPTS AND RULES

The origin algorithm use set to describe activities, such as $V=(a_1,a_2,...,a_n)$. The set can not denote the relationship between the sequences of time. And it can not describe duplicate tasks. Fox example, $(a_1, a_2, a_3) = (a_2, a_3, a_1)$, and $(a_1, a_2, a_3, a_3) = (a_1, a_2, a_3)$. We extend it based on sequence with time dimension. If take activities execution time as a sequence number such as N, we can define activities as follows.

Definition 1 $\text{seq V} = \{ \text{f} : (\text{T}, \text{V}) \mid \text{T} \subseteq \text{N}, \text{dom f} = 1..\#\text{V} \}$.

Due to V is a limited set and the #V=card(V), T will be a limited subset of N to denote corresponding activity time. For example, there are some activities $v_1=(a_1, a_2, a_3)$, their execution time set is (t_1, t_2, t_3), the seq $v_1=\{(t_1, a_1), (t_2, a_2), (t_3, a_3)\}$. In this way, the different order of activities and the duplicated activities can be described accurately.

The set of all activities in workflow logs is binary denoted as seq $V=((t_1, a_1), (t_2, a_2),..., (t_n, a_n))$. And the workflow instance S is a set of quadruples denoted as $(ID, (t_i,a_i), t_s, t_e)$, where InsID is ID of the instance. $(t_i,a_i) \in seq\ V$. The t_s and t_e are starting and ending time. S0 is the set of all instance in workflow logs. A certain activity set of instance I is denoted as $AS(I)=\{(t_i, a_i): (ID, (t_i,a_i), t_s, t_e) \in I\}$.

To discover workflow process models, the activities dependency relationships are been defined as follows.

Definition 2 In workflow instance set S, activity a_i depends a_j partially, denoted as $(t_i,a_i) \prec_s (t_j,a_j) \Leftrightarrow \forall I \in S, (t_i,a_i) \in seqV, (t_j,a_j) \in seqV \bullet t_i < t_j$.

This definition means in all instance set, a_i will start early than a_j. It is easy to proof that $(t_i,a_i) \prec (t_j,a_j)$ is transitive. For example if $(t_i,a_i) \prec (t_j,a_j)$ and $(t_j,a_j) \prec (t_k,a_k)$, then $(t_i,a_i) \prec (t_k,a_k)$. Due to this definition is based on all workflow log, this relationship needs completely logs data.

Definition 4 In workflow system, activity a_i depends a_j directly, denoted as $(t_i,a_i) \prec' (t_j,a_j) \Leftrightarrow \forall I \in S, \exists a_k, (t_i,a_i) \prec (t_k,a_k) \wedge (t_k,a_k) \prec (t_j,a_j)$.

Based on this algorithm's data structure, every workflow instance can be described by a extended directed diagram

978-0-7695-3927-0/09 $26.00 © 2009 IEEE
DOI 10.1109/ICRCCS.2009.73

which node is binary group of seq V. The relationship of $(t_i,a_i) \prec (t_j,a_j)$ is based on all workflow logs, so the algorithm will be complicated. Therefore $(t_i,a_i) \prec '(t_j,a_j)$ reduce the computing. $(t_i,a_i) \prec '(t_j,a_j)$ means in any workflow instance, there is a directed path starting from a_i through a_k end with a_j.

Definition 5 S is workflow instance set, $\forall I_i$, $I_j \in S$, $AS(I_i) = AS(I_j)$, then the dependency matrix corresponding to S is defined as $D = (d_{ij})$. If $(t_i,a_i) \prec (t_j,a_j)$, d_{ij}=1, else if $(t_j,a_j) \prec (t_i,a_i)$, d_{ij}= -1, else d_{ij}=0.

Where AS(I) means the workflow instance I's activities.

Based these definitions, reference [8] can only describe causality and single step loop relationship. To describe duplicated tasks, parallelism and select relationship, new definitions as follows are needed.

Definition 6 Duplicated tasks denoted as activity a_i and a_j , they have same name, but they maybe happened different time sequence in same instance.

Reference [9] considered concurrent activities relationship. The parallel tasks were divided into six categories: (a) sequence/sequence, (b) sequence/parallel, (c) sequence/select, (d) parallel/parallel, (e) parallel/select and (f) select/select. These relationships can be shown in figure 1.

The sequence/sequence relationship can be discovery by definition 3. Based pre-definitions, we proposed rules as follows to discovery other duplicated tasks relationships.

Rule 1 In workflow logs, If $(t_i,a_i) \prec (t_j,a_i)$, then always $\exists (t_i,a_i) \prec (t_k,a_k) \wedge (t_j,a_i) \not\prec (t_k,a_k)$. Task (t_i,a_i) and (t_j,a_i) have same activity name, but occurred repeatedly in different time, so they are duplicated tasks. Due to if (t_j,a_i) was executed, there always (t_k,a_k) was executed, so they have sequence/parallel relationship. For example in figure 1 chart (b), task X and task B are sequence/parallel relationship. The sequence/parallel relationship is denoted as $(t_i,a_i) \prec\!\!\triangleq (t_j,a_i)$.

Rule 2 In workflow instances set S $\exists (t_i,a_i) \prec_S (t_j,a_i)$, and in other instances set T $\exists (t_i,a_i) \prec_T (t_k,a_k) \wedge (t_j,a_i) \not\prec (t_k,a_k)$. Task (t_i,a_i) and (t_j,a_i) are duplicated tasks. Due to in some instance set (t_j,a_i) was executed, and (t_k,a_k) was executed in other instance set, so they have select relationship. For example in figure 1 chart (c), task X and task B are sequence/select relationship. The sequence/select relationship is denoted as $(t_i,a_i) \prec \oplus (t_j,a_i)$.

Rule 3 In workflow logs, If $(t_s,a_s) \prec (t_i,a_i)$, $(t_s,a_s) \prec (t_k,a_i)$, $(t_i,a_i) \not\prec (t_k,a_i)$,then always $\exists (t_s,a_s) \prec (t_j,a_j)$, $(t_s,a_s) \prec (t_l,a_j) \wedge (t_j,a_j) \not\prec (t_l,a_j)$. Task (t_i,a_i) and (t_k,a_i) are duplicated tasks. Task (t_j,a_j) and (t_l,a_j) are duplicated tasks. And activities named a_i and activities named a_j are parallel/ parallel relationship. For example in figure 1 chart (d), tasks named 'A' and tasks named 'B' are parallel/ parallel relationship. The parallel/ parallel relationship is denoted as $(t_i,a_i) \triangleq (t_l,a_j)$.

Rule 4 In workflow logs, if $(t_i,a_i) \prec \oplus (t_j,a_i)$ and $(t_j,a_i) \prec \oplus (t_i,a_i)$, then task (t_i,a_i) and (t_j,a_i) have parallel/select relationship. This relationship can be denoted as $((t_i,a_i) \prec \oplus (t_j,a_i)) \triangleq ((t_j,a_i) \prec \oplus (t_i,a_i))$. For example in figure 1 chart (e), tasks named 'A' and tasks named 'B' are parallel/select relationship.

Rule 5 In workflow logs, if $(t_i,a_i) \prec '\oplus (t_j,a_i)$, and $(t_j,a_i) \prec '\oplus (t_i,a_i)$, then task (t_i,a_i) and (t_j,a_i) have select/select relationship. This relationship can be denoted as $((t_i,a_i) \prec \oplus (t_j,a_i)) \oplus ((t_j,a_i) \prec \oplus (t_i,a_i))$. For example in figure 1 chart (f), tasks named 'A' and tasks named 'B' are select/select relationship.

The definition 6 can only identify activities' sequence relationship, this paper extend it with other five relationships based on rule 1 to rule 5.

Definition 7 S is workflow instance set, $\forall I_i$, $I_j \in S$, $AS(I_i) = AS(I_j)$, then the relationship matrix corresponding to S is defined as $R = (r_{ij})$. The r_{ij} 's value can been gotten reference the follow equation.

$$r_{ij} = \begin{cases} 1 & (t_i,a_i) \prec (t_j,a_j) \\ 2 & (t_i,a_i) \prec\!\!\triangleq (t_j,a_i) \\ 3 & (t_i,a_i) \prec \oplus (t_j,a_i) \\ 4 & (t_i,a_i) \triangleq (t_l,a_j) \\ 5 & ((t_i,a_i) \prec \oplus (t_j,a_i)) \triangleq ((t_j,a_i) \prec \oplus (t_i,a_i)) \\ 6 & ((t_i,a_i) \prec \oplus (t_j,a_i)) \oplus ((t_j,a_i) \prec \oplus (t_i,a_i)) \\ 0 & others \end{cases}$$

Definition 7 not only define the sequence relationship, but also present the other five relationships between tasks with same name or different.

III. MINING PROCESS TO FIND DUPLICATED TASKS

The origin algorithm in reference [8] considers the directed partial order to build activity directed diagram. The origin algorithm can be briefly summarized as:

1) The workflow instances are classified into some similar set S_i based on time sequence features.

2) Calculate each S_i 's dependency set D_i based on definition 5.

3) Calculate directed dependency relation matrix for each D_i by using CompDirDepend algorithm.At this time, the preliminary workflow model was found. In this model, some sequence and parallel canbe identified, but can't get select relationship between activities.

4) Discovery select relationship using CompEDG algorithm.

This mining process is effective for workflow models without duplicated tasks. This paper extends these steps to discovery duplicated tasks and other complex activity relationships.

In order to set out a better understanding of the algorithm, we take a workflow model with duplicated tasks to be a presentation. The model is showed in figure 2.

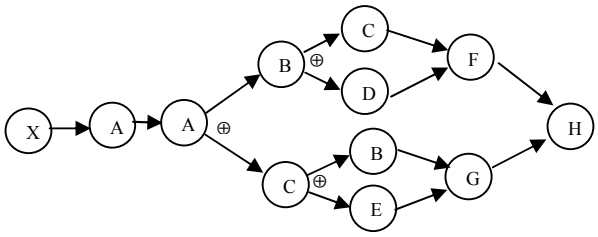

Figure 1. Example of Workflow Model

This workflow model contains most of relationships between duplicated tasks.

A. Data Preprocessing

In this paper, workflow tasks are denoted based on definition 1 to identify duplicated tasks.

Classify workflow instance set into similar set S_i using method in reference [8]. Then we can find instance I in S_1 through the path (X, A, A, B, C, F, H) and the path (X, A, A, C, B, G, H). And the instance I in S_2 through the path (X, A, A, B, D, F, H) and the path (X, A, A, C, B, G, H). The instance I in S_3 through the path (X, A, A, B, D, F, H) and the path (X, A, A, C, E, G, H). The instance I in S_4 through the path (X, A, A, B, D, F, H) and the path (X, A, A, C, E, G, H).

B. Calculate Relationship Matrix

For each S_i in step A, we can calculate their corresponding relation matrix based definition 7 and rule 1 to rule 5. Then we can calculate R_1, R_2, R_3 and R_4 quickly. Here the value of R_1, R_2, R_3 and R_4 are omitted.

$$R_1 = \begin{array}{c} \\ (t_x, X) \\ (t_{a1}, A) \\ (t_{a2}, A) \\ (t_{b1}, B) \\ (t_{c1}, C) \\ (t_d, D) \\ (t_f, F) \\ (t_{c2}, C) \\ (t_{b2}, B) \\ (t_e, E) \\ (t_g, G) \\ (t_h, H) \end{array} \begin{pmatrix} & & & & & & & & & & & \\ 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 1 & 2 & 2 & 0 & 2 & 2 & 2 & 0 & 2 & 1 \\ 0 & 0 & 0 & 2 & 2 & 0 & 2 & 2 & 2 & 0 & 2 & 1 \\ 0 & 0 & 0 & 0 & 5 & 0 & 1 & 5 & 0 & 0 & 2 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 5 & 0 & 5 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 5 & 5 & 0 & 5 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 5 & 0 & 4 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{pmatrix}$$

As can be seen in matrix R_1, column sixth and column tenth are filled with zero. Similarly, in R_2 column fifth and column tenth are filled with zero. This is because the workflow instance in S_1 do not execute task (t_d, D) (column sixth) and task (t_e, E) (column tenth). Similarly, workflow instance in S_2 do not execute task (t_c, C) (column sixth) and task (t_e, E) (column tenth). Task (t_c, C) and task (t_d, D) are parallel/select relationship.

Therefore, the method of discovery global dependency relationship does not apply to the situation with duplicated tasks.

C. Calculate global decency matrix

Because the select relationship exists in duplicated tasks, the R_i can not indentify the whole relations alone. We need to analyze all R_i to find global relations. The Extended-CompDepend algorithm is proposed to find them.

CompDepend($(R_1, R_2, ..., R_i)$)

{ R_0 =ZeroMatrix(#S, #S);

/* firstly, find sequence dependency */

Initial R_0 by CompDirDependy algorithm in reference [8];

/* find sequence and select relations /*

For (i=0; i<#S, i++)

{ *for* (j=0; j<#S, j++)

{ (t_i,a_i) and (t_j,a_j) are R_0 (i, j)'s corresponding activity;

If (in all R_i , R_i(i, j) have same value)

{ R_0(i, j)= R_0(i, j);}

} }

/* find parallel dependency based rule 3 */

For (i=0; i<#S, i++)

{ *for* (j=0; j<#S, j++)

{ (t_i,a_i) and (t_j,a_j) are R_0 (i, j)'s corresponding activity;

If $(t_i, a_i) \nprec (t_j, a_j)$, and $\exists (t_s, a_s)$

$\bullet \{(t_s, a_s) \prec (t_i, a_j)$ and $(t_s, a_s) \prec (t_j, a_j)\}$

{ R_0 (i, j)=4;}

} }

/*find other relationships */
For (i=0; i<#S, i++)
{ *for* (j=0; j<#S, j++)
{ (t_i,a_i) and (t_j,a_j) are R_0 (i, j)'s corresponding activity;
If ((t_i,a_i) and (t_j,a_j) have the same task name)
{ *if* ($(t_i, a_i) \prec (t_j, a_i)$
and in all R_i $\exists R(i,x) = 4$)
{ R_0 (i, j)=2; break;}
if ($(t_i, a_i) \prec (t_j, a_i)$, $R_x \subset (R_1, R_2, ..., R_i)$
and in R_x $\exists R(i,x) = 4$)
{ R_0 (i, j)=3; break;}
} } };
Return R_0 ;
}

The R_0 value of this example is calculated as show in follows. The R_0 reveal the rich relationships between the various tasks. For example, task (t_{a1}, A) and (t_{a2}, A) are sequence/sequence relations. Task (t_e, E) and (t_{b1}, B) are select/select relations, etc.

Based on R_0 and R_i, the origin CompEDG method can construct activity model basically with directed diagram. The preliminary model contains sequence and parallel relations.

$$R_0 = \begin{array}{c} \\ (t_x, X) \\ (t_{a1}, A) \\ (t_{a2}, A) \\ (t_{b1}, B) \\ (t_{c1}, C) \\ (t_d, D) \\ (t_f, F) \\ (t_{c2}, C) \\ (t_{b2}, B) \\ (t_e, E) \\ (t_g, G) \\ (t_h, H) \end{array} \begin{array}{c} \begin{array}{cccccccccccc} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 & 11 & 12 \end{array} \\ \left\{ \begin{array}{cccccccccccc} 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 1 \\ 0 & 0 & 0 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 2 & 1 \\ 0 & 0 & 0 & 0 & 5 & 3 & 1 & 5 & 0 & 5 & 2 & 1 \\ 0 & 0 & 0 & 0 & 0 & 5 & 1 & 0 & 5 & 5 & 5 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 5 & 5 & 5 & 5 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 5 & 5 & 4 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 2 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 5 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{array} \right\} \end{array}$$

According to the following steps, the duplicated tasks relations can be added.

1) Select the first node in diagram and denoted as CurAct;

2) LeftSet = AS(I) – CurAct;

3) while($\text{LeftSet} \neq \varnothing$*){*

4) if the CurAct's corresponding value equals 2 in R_i*, then marker CurAct's subsequence edge with '〉';*

5) if the CurAct's corresponding value equals 3 in R_i*, then marker CurAct's subsequence edge with '⊕';*

6) if the CurAct's corresponding value equals 4 in R_i*, then marker CurAct's subsequence edge with '≙';}/*end of while */*

After filling with new relationships of duplicates tasks, the constructed model is integrity.

IV. CONCLUSIONS

A new method to identify and discover duplicated tasks based on directed diagram was proposed. The paper use sequence set of tasks to indentify the tasks with same name. And based partial order (proposed in reference [8]), the paper design five rules to find six relations between duplicated tasks. And then, an Extended-CompDepend-Algorithm based these rules was proposed to discovery tasks global relationships. At last, the steps to indentify these relationships in directed diagram was detailed. This paper focuses on discovering of duplicated tasks. To solve the multistep loop in the workflow model will be focus of the future research.

REFERENCES

[1] Gu, Chunqin; Chang, Huiyou; Yi, Yang; Overview of Workflow Mining Technology Granular Computing, 2007. GRC 2007. IEEE International Conference. 2-4 Nov. 2007 Page(s):347 - 347

[2] Weixiang Sun; Tao Li; Wei Peng; Tong Sun; Incremental Workflow Mining with Optional Patterns Systems. SMC '06. IEEE International Conference. Volume 4, 8-11 Oct. 2006 Page(s):2764 - 2771.

[3] Gaaloul, W.; Alaoui, S.; Baina, K.; Godart, C.; Mining Workflow Patterns through Event-Data Analysis. Applications and the Internet Workshops, 2005. Saint Workshops 2005. 31-04 Jan. 2005 Page(s):226 - 229.

[4] Aalst W M P,Weijters A J M M,Maruster L.Workflow Mining:Discovering Process Models from Event Logs[J].IEEE Transactionson Knowledge and Data Engineering,2004,9(12):369-378.

[5] Ye Mao, Zhao WeiDong, A new workflow mining algorithm. Computer Integrated Manufacturing System . 2006,11

[6] Cook J E.Process Discoverying and Validation Through Event-data Analysis[R].Boulder,University of Colorado,Technical Report: CU-CS-817-96,1996-11.

[7] Weidong Zhao, Ye Mao. A Workflow Frequent Pattern Mining Algorithm. 2007:

[8] De Medeiro A K A, van der Aalst W M P,Weijters A J MM. Workflow mining: current status and future directions//Meersman R, Tari Z, Schmidt DC. On the Move to Meaningful Internet systems 2003:CoopIS, DOA, and ODBASE. Berlin: Spring-Verlag, 2003:389-406