A Frequency-Based Algorithm for Workflow Outlier Mining

Yu-Cheng Chuang[1], PingYu Hsu[1], MinTzu Wang[2], and Sin-Cheng Chen[1]

[1] Department of Business Administration, National Central University
tocasper@hotmail.com, pyhsu@mgt.ncu.edu.tw,
93441024@cc.ncu.edu.tw
[2] Department of Information Management,
Technology and Science Institute of Northern Taiwan
mtwang@tsint.edu.tw

Abstract. The concept of workflow is critical in the ERP (Enterprise Resources Planning) system. Any workflow that is irrationally and irregularly designed will not only lead to an ineffective operation of enterprise but also limit the implementation of an effective business strategy. The research proposes an algorithm which makes use of the workflow's executed frequency, the concept of distance-based outlier detection, empirical rules and Method of Exhaustion to mine three types of workflow outliers, including less-occurring workflow outliers of each process (abnormal workflow of each process), less-occurring workflow outliers of all processes (abnormal workflow of all processes) and never-occurring workflow outliers (redundant workflow). In addition, this research adopts real data to evaluate workflow mining feasibility. In terms of the management, it will assist managers and consultants in (1) controlling exceptions in the process of enterprise auditing, and (2) simplifying the business process management by the integration of relevant processes.

Keywords: ERP, BPM, Workflow mining, Data mining, Outlier detection.

1 Introduction

The current highly competitive business environment is comprised of global markets, changeable demands of customers and a ferocious competition. Enterprises must strengthen inside operations to maintain a competitive advantage, with a view of making an enormous profit.

The whole business operating strategy ought to be firmly grounded on the utmost maximization of the competence in profiting and increasing the revenue of the enterprise and the satisfaction of the customers. In order to fulfill this, the position of the customers and the ways to gratify them shall be taken into consideration in the process of business organization. Meanwhile, these elements need to be integrated into the whole business operating processes from customers to suppliers [6]. In view of this, many enterprises use information systems like ERP、SCM、CRM、B2B and WfMS to execute or integrate all business processes. These information systems will keep track of the processing details, which are presented as system logs.

T.-h. Kim et al. (Eds.): FGIT 2010, LNCS 6485, pp. 191–207, 2010.

Workflow management system (WfMS), such as Staffware, IBM MQSeries, COSA, based on the definition of WfMC (The Workflow management Coalition), is a system to define, manage and implement workflow. The design of the system must present the logic of workflow completely.

Similar to the concept of a workflow management system, Business Process Management (BPM) is defined as the software and tool to establish and implement the business workflow model by dint of the definition and integration of any necessary personnel, system, application and applied unit, according to the Butler Group [20]. Four major parts are included in BPM: (1) The analysis and construction of the workflow,(2) workflow management,(3) the application and integration of the business, and (4) the surveillance and management of the workflow.

In addition, the implementation of ERP is deemed as one of the pivotal issues. With reference to the example of ASAP (i.e. the implementation method of SAP), the implementation can be divided into five stages, including (1) project preparation, (2) business blueprint, (3) system realization, (4) system final preparation and (5) system co-live & continuous improvement. Consequently, workflow outlier mining is instrumental in terms of the execution of the analysis and construction of the workflow, which in turns helps BPM impeccably integrate the business processes into the information system and improves the efficiency of the execution of ERP and modify the system variables to fine-tune scenarios in the stages of business blueprint and system realization beforehand.

In terms of the management, it can assist managers and consultants in:

1. Managing the exception effectively in the process of internal control and auditing.
2. Simplifying business operating processes by dint of the integration of pertinent processes. For instance, the installation of the BPM system is conducive not only to the analysis of the workflow and the execution of the construction, with a view to producing optimal integration of business processes and the information system, but also to adjust system variables in the process of the installation of ERP by the omission of unnecessary steps of workflow to optimize the system.

By the way, most previous researches pertaining to workflow are focusing on how to reconstruct the workflow process by system log and very few of them are to mine workflow outlier. According to the motivation for this research, one objective of this research is to excogitate an algorithm suitable for every system log for mining less-used or never-used business processes. The features of system logs will be the primary focus to probe into the workflow outlier by means of executed process frequency and statistical methods. In addition, the Method of Exhaustion will be adopted to find out the never-occurring workflow outliers.

The remainder of this paper is organized as follows. Section 2 is mainly with literature review, offering references pertinent to workflow mining and outlier detection. The process model and the algorithm for mining adopted in this research will be elucidated in Section 3, which demonstrates the methodology and the algorithm of this research. The implementation of the algorithm as a rudimental system will be conducted in Section 4 to assess the accuracy of this algorithm, while the results of the research and the direction of future researches will be summarized in Section 5.

2 Literature Review

The pertinent references, including (1) workflow mining and (2)outlier detection. The research of Process Discovery has always been the most prevalently discussed in the realm of workflow mining (see Fig.1).

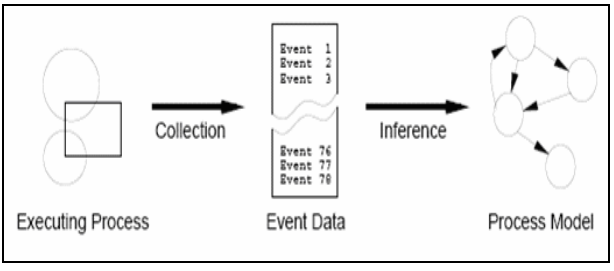

Fig. 1. Process Discovery

2.1 Workflow Mining

In Cook & Wolf's research [13], they suggest three methods for process discovery, inclusive of neural networks, as the purely algorithmic approach and the Markov approach the authors consider being the most promising. On one hand, the purely algorithmic approach establishes a finite state machine (FSM) to connect any similar recurrence in the future, which produces the method for the final process model. On the other hand, the Markov approach adopts a mixture of algorithmic and statistical methods, having the ability to deal with noise. This research is conducted within a sequential behavior.

Cook and Wolf extend their study to concurrent processes [14]. They devise specific metrics (entropy, event type counts, periodicity, and causality) and use these metrics to discover models out of event streams. Yet, they do not provide any approach productive of specific process models.

The idea of the application of process mining to the real in the context of workflow management was first introduced in [19], which is grounded on workflow graphs under the inspiration of workflow products like IBM MQSeries workflow (formerly known as Flowmark) and InConcert. In this paper, two problems are defined. The first problem is to find the occurrence of events that a workflow graph generates in a given workflow log. The second problem is to find the definitions of edge conditions. A concrete algorithm is served as a means to tackling the first problem. The approach differs considerably from the others by reason of the nature of workflow graphs; therefore, it is needless to identify the nature (AND or OR) of merges and splits. Although in terms of the handling of a partial situation of recursion, Method of Exhaustion is adopted and the potential graphs are adjusted in this paper, the process model is incompletely presented.

Schimm [5] has developed a mining tool suitable for discovering hierarchically structured workflow processes. This requires the maintenance of the equilibrium of all splits and merges.

Herbst and Karagiannis also address the issue of process mining in the context of workflow management by means of an inductive approach. The approach presented respectively in [8][9][10][11][12] allows for concurrency, using stochastic task graphs as an intermediate representation and generating a workflow model described in the ADONIS modeling language. In the process of induction, task nodes are merged and divided in order to discover the underlying process. A notable difference from other approaches is that the same task node can appear multiple times in the workflow model. In [1] & [2], a heuristic approach rather than simple metrics is used to construct so-called "dependency/frequency tables" and "dependency/frequency graphs".

Among the researches pertaining to workflow mining, most of them concentrate on the process discovery of workflow mining. However, in [18], the researchers mine a process model able to adjust configures by each activity's execution frequency. It is pointed out in this research that the application of the concept of workflow mining to Information System such as ERP, CRM and SCM will contribute abundant commercial benefit to the enterprise and fortify its competitiveness. Hence, how to detect and mine the workflow outlier is another important focus for workflow mining.

2.2 Outlier Detection

At present, the bulk of studies relevant to outlier detection are categorized as a part of statistics. At this time an "outlier", or "noise", receives no categorically unanimous definition, Hawkins proffers a definition inclusive thoroughly of the characteristics and the pith of an outlier, indicating that, "An outlier is an observation that it was generated by a different mechanism [7]." Outlier detection is one of the fundamental issues in data mining, especially in fraud detection, network intrusion detection, network monitoring, etc. The traditional methods of outlier detection are classified as follows:

(1) Distribution-based methods in this category were previously conducted by the statistics community. Some standard distribution models (e.g. normal) are employed in these methods and those points that deviate from the model as outliers are flagged. Recently, Yamanishi, Takeuchi and Williams [23] used a Gaussian mixture model (GMM) to present that normal behaviors and each datum is scored based on changes in the model. A high score likely manifests a high possibility as an outlier.

This approach has been in combination with a supervision-based learning approach to obtain general patterns for outliers [22]. In view of arbitrary data sets without any prior knowledge of the distribution of points, conducting expensive tests to determine which model fits the data best, if any, is ineluctable.

In 2002, Yamanishi and Takeuchi suggest an integrated framework to cope with both of them on the basis of the theory of the on-line learning of non-stationary time series [21].

(2) Deviation-based techniques ascertain outliers by inspecting the characteristics of objects and deem an object that diverges from these characteristics as an outlier [3].

(3) Distance-based methods are originally proposed by Knorr and Ng [15][16][17]. This notion generalizes many ideas from the distribution-based approach and renders better picture of computational complexity.

(4) The Density-based method is proposed by Breunig, Kriegel, Ng, and Sander [4]. It relies on the local outlier factors (LOF) of each point in accordance with the local density of its neighborhood. From the typical use, points with a high LOF are flagged as outliers.

In conclusion, as far as this research is concerned, the activity's execution frequency and a distance-based method for Workflow Outlier Mining will be adopted. After discovering the workflow outlier, this research also makes use of process reconstructing redundant process from never occurring workflows.

3 Methodology

The concepts pertinent to workflow will be introduced in this section. In addition, the algorithm for workflow outlier mining will be designed in accordance with the characteristics of workflow.

3.1 Related Workflow Definitions

According to the definition of Workflow from Workflow Management Coalition (WfMC), here are relevant concepts of workflow:

Definition 1: Workflow is also known as Process, which is to connect predefined rules and transmit the document, information or task among participants of the process.

Definition 2: Activity is also known as "Job" or "Task", which is used to describe one task in a workflow. In logical terms, it can be seen as the smallest complete task. Not all of the activities, which can be divided into either manual or automatic activity, are executed automatically. The manual activity is indicative that the workflow execution should be executed under the activity participant's assistance or supervision. The automatic activity means that the activity is executed by the trigger program and works independently.

Definition 3: Instance is the actual situation of workflow execution. It is feasible that many Instances are running simultaneously without interfering with each other.

Generally speaking, workflow can be judged from two points of view: Design Time and Run Time. From the perspective of Design Time, all the workflows are predefined. Take Fig.2 for example. Many Processes (P_1, P_2, P_3…P_n) are involved in a system, and each process is comprised of many Activities (A_1, A_2, A_3…A_n), each of which has its own type. Take the Purchase Order Process for example. A_1 may be the START type, indicative of the activation of a purchase order; A_2 may be the WEB type and it means a purchaser list that contains all required materials; A_3 may be the XOR type, representing IS checks the rationality of material specification; A_4 maybe the SUBFLOW type, standing for the following procedure of purchase order; A_5 may be the END type, representing the end of this process.

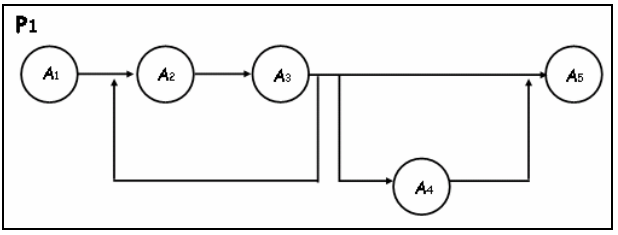

Fig. 2. The Concept of Process

From the viewpoint of Run Time, the record of execution named "Instance", which is comprised of WorkItem. As shown in Fig.3, Process P_1 has been executed for four times in a real situation, leaving the execution record Instance I_1, I_2, I_3, I_4. The actual execution situation is demonstrated as follows: I_1's execution sequence is W_1, W_2, W_3, W_5; I2's execution sequence is W_1, W_2, W_3, W_4, W_5; I_3's execution sequence is W_1, W_2, W_3, W_2, W_3, W_5; I_4's execution sequence is W_1, W_2, W_3, W_2, W_3, W_4, W_5.

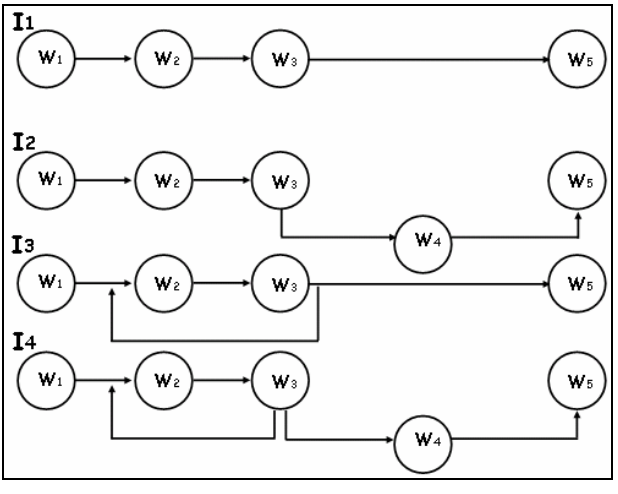

Fig. 3. The Concept of Instance

3.2 The Algorithm for Workflow Outlier Mining

In this section, the application of the concepts of workflow outlier to the workflow will be conducted, and the algorithm for workflow outlier mining will be contrived in accordance with the characteristics of workflow. The task for workflow outlier mining will be divided into two parts: One is to search for the less-frequent abnormal workflow appearing in the system, and the other aims at the never-executed redundant workflow in the system.

According to the Empirical Rule of Statistics, the probability of data located at the range [X-S, X+S] is 68%; the probability of data located at the range [X-2S, X+2S] is 95%; the probability of data located at the range [X-3S, X+3S] is 99.7%. Based on the

motivation of the research mentioned previously, it will provide consultants and managers with valuable information if less-occurring workflow (also called Abnormal Workflow) can be mined from Instance, which represents the actual execution situation.

Consequently, the concept of FPOF (Frequent Pattern of Factor) [24] will be ameliorated into FIOF (Frequent Instance of Factor), which is suitable for workflow outlier mining. FIOF is used to measure the outlier's degree of Instance and able to form the data range in combination with Empirical Rule, and finally workflow outlier will be mined.

Abnormal Workflow can be divided into two types. The first type is less-occurring instance in each process. Instance's Process Activities sequence is Abnormal Workflow. Take Fig.4 for example. In the actual execution records, Instance I1's frequency is twice. Apparently it is a kind of less-occurring workflow, and its Process Activities sequence (A1, A2, A3, A8) is the workflow outlier.

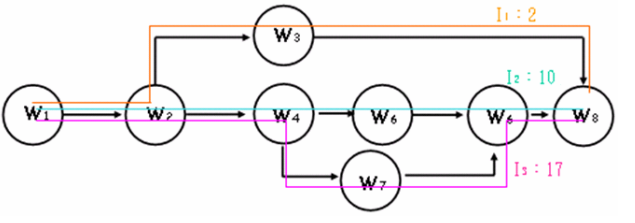

Fig. 4. Less-occurring workflow

In terms of the first type of workflow outlier, an algorithm called "Abnormal WMe" (Abnormal Workflow Mining for each process) is designed, and the procedure is shown as follows:

(1) Take all average numbers of the Instance (in each Process) support as minimum support (min_support).
(2) Find out the number of the Instance support larger than the min_support, which is called frequent instance in each Process. The number of frequent instance is called ||FIS (P, min_support)||. FIS (Frequent Instance Set) means the set of instance in which frequency is higher than the threshold of min_support.
(3) Calculate each Instance's FIOF (Frequent Instance of Factor).

FIOF(i) = support(i)*||FIS(P, min_support)||

(4) Use the Empirical Rule to calculate the probability range of FIOF. Those Instances' FIOF numbers located at the left side of range are Abnormal Workflow.

The point of this algorithm, is that after the calculation of the frequency of each Instance, the instance support's average number will be viewed as min_support, which represents the trend this data set focuses on. If an Instance's support is lower than min_support, it is likely to be an outlier. Then calculate each Instance's FIOF, and use the Empirical Rule to find out relatively less-occurring Instance, of which Process Activities sequence is Abnormal Workflow.

In step 3- the calculation of FIOF, the purpose of being multiplied by ||FIS(P, min_support)|| is to emphasize Instance's degree of outlier if its Process has fewer frequent instances. As shown in Fig.5, Process a and Process b both have an Instance's support that is 2/11=0.182. It is probably inferred that both of them are outliers. But it is observable that in Process b, Ib2's frequency (2) is far less than Ib1's frequency (9). In Process a, though Ia3's frequency is also 0.182, it is not far different from other Instances Ia1 and Ia2. Hence, being multiplied by ||FIS(P, min_support)|| can underscore the degree of outlier, and as a result, Ib2 has higher probability to be outlier than Ia3 (0.182<0.364).

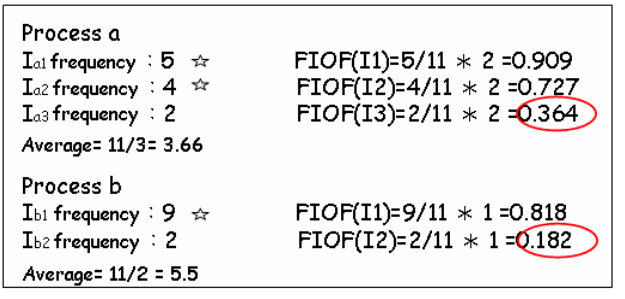

Fig. 5. The Example of How ||FIS(P, min_support)|| Affects FIOF

The detailed algorithm AbnormalWMe is displayed as Fig 6:

The second type of Workflow Outlier is the less-occurring Process in all Processes relatively. The procedure is shown as follows:

(1) Sort all Processes by their total Instance number, and take all Process' support average as min_support.
Use the Empirical Rule to find out the Workflow Outlier located beyond the range.

Algorithm 1. AbnormalWMe (abnormal workflow mining for each process)

Input： P, all Process' set, $P= (p1,p2,p3…pi…pn)$;
$I(pi)$,each Process' Instance's set, $I(pi)=(i1,i2,i3…ii…in)$;
S, all Instance's support's set, $S=(s1,s2,s3…si…sn)$;

Output： Abnormal workflow set AW,

for each Process pi **do**
min_support(pi)= Average number of Support;
//Find each Process' min_support。
FIS_number(pi)
//Find how many frequent instances are larger than min_support，||FIS(P,min_support)||。

Fig. 6. The AbnormalWMe Algorithm

```
{
 int number=0;
  for (i=1; i<=n; i++)
   if (si >= min_support) FIS_number=FIS_number+1;
   else return FIS_number;
  return FIS_number;
 //Search from the first instance to the last one. If the instance support is larger than
    min_support, add the number.
}

AbnormalWMe(P)
//Mine less-occurring workflow in each Process, and its Process Activities
sequence is the workflow outlier
{
  for each Instance ii do
    FIOF(ii)= si * FIS_number(pi)
     //Calculate each instance's FIOF(ii)=support(i) * ||FIS(P,min_support)||
    Calculate X(arithmetic mean of FIOF(I))and S(standard deviation of FIOF(I))
       //Calculate all Instances' FIOF average X and standard deviation S
   let lower_bound=X-S;
   //According to the Empirical Rule, there is 68% data which lower bound is X-S
   for each FIOF(ii) do selection_sorting();
   //Use selection sorting to sort all Instances from small FIOF to large FIOF
     for (ii=1; ii< n; ii ++)
      if (FIOF(ii)<=lower_bound) return AW[ii] ;
      else return 0;
     return 0;
   //Start from the smallest FIOF. If FIOF is lower than lower_bound, return this
     Instance's Process Activities sequence. These left range outliers are
     Abnormal Workflow.
}
```

Fig. 6. (*continued*)

3.3 Redundant Workflow Mining

The other type of Workflow Outlier consists of those Processes' Instances that never-occur in system logs. It is possibly because there are many probably-occurring sequences (Instance) in a Process, while some sequences are not suitable in a real scenario. Hence, some sequences are never-occurring. Regardless of BPM Project or ERP Implementation, finding out these never-occurring sequences will improve the enterprise's operating efficiency.

The concept is to use Method of Exhaustion to find out every Process' potential execution sequence and subtract with Instance (Process Activities sequence that really occur) to mine the never-occurring Workflow.

As demonstrated in Fig. 7(a) through Fig. 7(c):

(1) The first graph is a Process' original model. The nearest split type Activity shall be searched for from the last Activity.
(2) In the second graph, the nearest split type Activity E is found. Start from E to build Activity Path. Use E[2] to present two paths of E: (EJLN) and (EKMN).
(3) In the third graph, the next nearest split type Activity G is discovered. Start from G to build Activity Path. Use G[3] to present three paths of G: (GDE[2])and (GHIKMN). It means (GDEJLN), (GDEKMN) and (GHIKMN).
(4) In the fourth graph, the last split type Activity A is detected, which is the start of the Process. Start from A to build Activity Path, and thereby use A[5] to present five paths of A: (ABCDE[2]) and (AFG[3]); that is, (ABCDEJLN), (ABCDEKMN), (AFGDEJLN), (AFGDEKMN), and (AFGHIKMN). Because Activity A is this Process' start type Activity, the five paths are all possible Activity sequence. This means Set(P)={ (ABCDEJLN)、(ABCDEKMN)、(AFGDEJLN)、(AFGDEKMN)、(AFGHIKMN)}
(5) In the fifth graph is the Instance (ever-occurred Process Activity sequence) in system logs.

Instance(P)={ (ABCDEJLN) 、(AFGDEKMN) }

(6) The sixth graph is representative of the subtract of Set(P) and Instance(P), which is the never-occurring Workflow in the system. It includes { (ABCDEKMN)、(AFGDEJLN) 、(AFGHIKMN) }

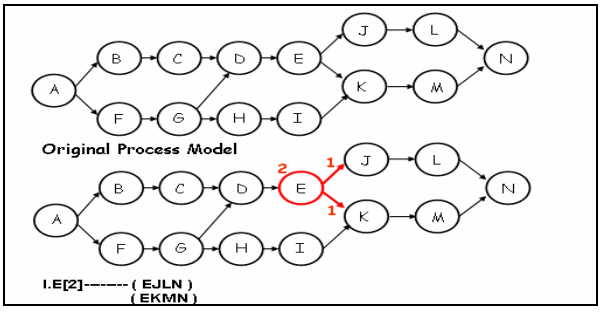

Fig. 7a. The Procedure of Mining Redundant Workflow

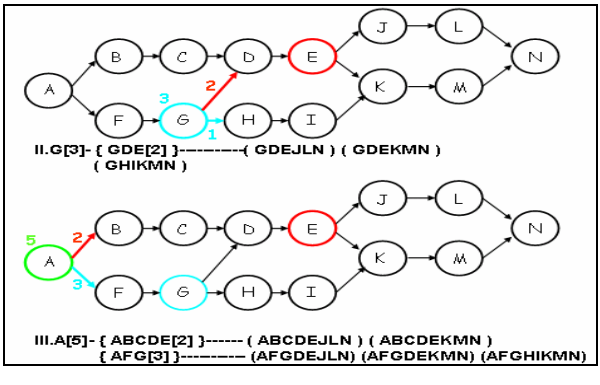

Fig. 7b. The Procedure of Mining Redundant Workflow

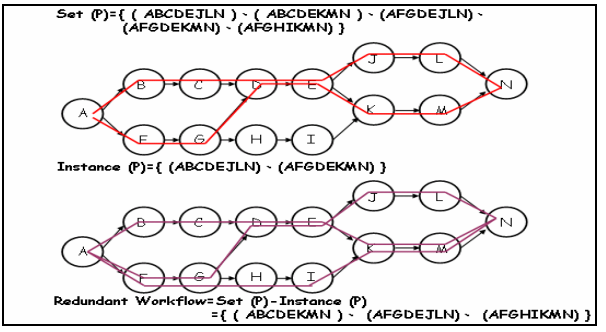

Fig. 7c. The Procedure of Mining Redundant Workflow

4 Experiments and the Evaluation

This section is divided into two parts. The methods for the design of experiments will be elucidated in the first part, while the results of experiments will be further explicated and analyzed in the second part.

4.1 The Design of Experiments

This experiment is designed to utilize the real Workflow System Log to validate the feasibility of algorithm, with a view to mining three discrepant types of Workflow Outliers, including less-occurred Workflow in each Process, less-occurred Workflow in all Processes, and never-occurred Workflow in each Process.

The procedure of the experiment is shown as follows:

(1) Dump workflow system logs into the database.
(2) Calculate each Process' helpful figures like the support of each Process and that of each Instance.
(3) Find out all Workflow Outliers in advance to compare the accuracy of algorithm according to the respective definitions of three Workflow Outliers.
(4) Put Workflow Log into the experimental system to mine Workflow Outliers.
(5) Compare the result with identified Workflow Outliers to evaluate the accuracy of algorithm.

In respect to the real data, it emanates from the backup archive stored in certain domestic manufacturing company's workflow management system Log. The record Instance contains the information over six months (from July, 2006 to December, 2006). There are twenty-five Predefined Process Models, only nine ever-occurred Processes exist. It is probably because the data source records for six months only while many Processes still continue running. There are about thousands of instances, however; only 68 among them are finished. It means the other Workflows are not finished but still running. Nevertheless, in this research, only Complete Workflows

are focused on to mine Workflow Outlier. As a consequence, the data amount is tenfold to six hundred and eighty Complete Workflows which generated by discrete distribution with proportional probability to simulate actual data volume. Based on the confidential contract with the cooperative company, all the Workflow names have been changed.

Besides, to make the algorithm more efficient, adjusted metadata is shown as follows:

- ✓ **Process(DEFINITIONID、ACTIVITYNAME、ACTIVITYTYPE、SPLIT_NEXT)**

DEFINITIONID represents each Process that includes many Activities; ACTIVITYNAME stands for each Activity's name; ACTIVITYTYPE means each Activity Type, including START, SPLIT and END; SPLIT_NEXT is a newly added field and data, which is used to record this SPLIT type Activity's next connected ACTIVITYNAME. This data structure will conduce to the ideal performance of the algorithm.

- ✓ **Instance(INSTANCEID、ACTIVITYNAME、ACTIVITYTYPE)**

INSTANCEID is indicative of each Instance; ACTIVITYNAME and ACTIVITYTYPE record Activity information, where the Instance is executed through.

From the observation of these Instances, some Workflow executed loops appear more than once. The objective of Workflow Outlier Mining is to find out Workflows that don't happen frequently. If any execution frequency of loop appears more than once, it will be classified as the same one. As shown in Fig.8, I_3 and I_4 of this Process are executed twice and also three times separately in (W_2, W_3). These situations are classified as I_2, the loop that is executed only once.

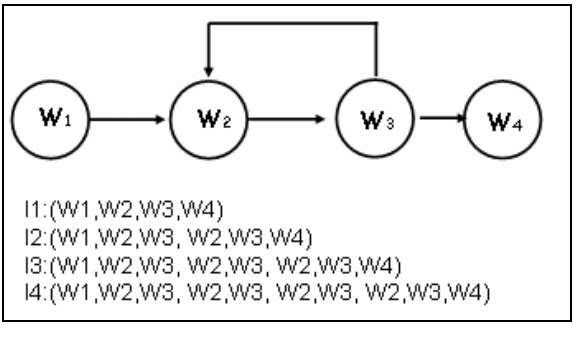

Fig. 8. How to deal with Loop Iteration

Furthermore, in this research, the Process model representative of the connection of Activities has been comprehended beforehand, which will assist the algorithm in building Process Activities sequence in the process of dealing with Redundant Workflow.

4.2 The Result and Analysis of Experiments

Experiment 1- AbnormalWMe

Experimental Object: Mine the less occurred Instance in each Process, of which Process Activities sequence is deemed as Abnormal Workflow.

Calculate each Instance's support, and use the support average number as min_support. Calculate ||FIS(P, min_support)|| (how many Instance's supports are larger than min_support), and calculate FIOF. All FIOF's average number X is 0.652; standard deviation S is 0.369; X-S is 0.283. Consequently, Workflow Outliers should be those FIOF smaller than 0.283. The extra Outliers are the Process Activities sequence of these two Instances- SO_02 and SO_04 as Fig.9 shows.

If without FIFO, we may get different results such as SOSub_01, SOSub_07. After the observation of these Instances, Process SO's Instance execution frequencies are 10, 21, 49 and 70. It is perceived clearly that 10 and 21 Instance execution frequencies are significantly lower than 49 and 70.

Process	execution	median	Instance	execution	support	||FIS(P,min_support)||	FIOF		
HFUMain_02	9		HFUMain_01_I10_04	9	1	1	1		
PPC_01	36		PPC_01_I40_04	36	1	1	1		
Pmain_04	82		PMain_01_I80_06	82	1	1	1		
Psub_12	88	20	PSub_05_I10_07	10	0.11364	3	0.340909		
			PSub_03_I10_07	9	0.10227	3	0.306818		
			PSub_01_I20_06	20	0.22727	3	0.681818		
			PSub_02_I20_07	19	0.21591	3	0.647727		
			PSub_04_I30_06	30	0.34091	3	1.022727		
RBFT_02	10		RBFT_01_I01_06	10	1	1	1		
SO_04	150	35	SO_04_I10_06	10	0.06667	2	0.133333		
			SO_02_I20_07	21	0.14	2	0.28		
			SO_01_I50_05	49	0.32667	2	0.653333		
			SO_03_I70_04	70	0.46667	2	0.933333		
SOSub_190	235	20.5	SOSub_01_I10_12	11	0.04681	4	0.187234		
			SOSub_07_I10_14	10	0.04255	4	0.170213		
			SOSub_04_I20_24	21	0.08936	4	0.357447	X =	0.652173913
			SOSub_08_I20_14	19	0.08085	4	0.323404	S =	0.368904952
			SOSub_03_I20_06	21	0.08936	4	0.357447	X-S=	0.283268961
			SOSub_06_I20_16	20	0.08511	4	0.340426		
			SOSub_05_I50_10	51	0.21702	4	0.868085		
			SOSub_02_I80_07	82	0.34894	4	1.395745		
RT_06	10		RT_01_I10_13	10	1	1	1		
VV_01	60		VV_01_I60_05	60	1	1	1		

Fig. 9. The result of AbnormalWMe

Experiment 2- AbnormalWMa

Experimental Object: Mine less-occurred Process in all Process set.

Finding out the less-occurred Process in all Process set is essential, whereas in the whole twenty-five Process Models, only nine ever-occurred Processes exist. This situation will give rise to the bulk of the support numbers that are 0 in the data set. In terms of the frequency-based concept, it will be unproductive of any result.

To supplement the insufficiency of data volume, the random number table is used to simulate all twenty-five Process' Instance numbers. The twenty-five Process' execution frequencies are shown as follows: (CRFT:292,AT:803,EXF:518, HFS:902,MT:925,RP:319,RPI:037,RPIA:407,RCFT:109,SDAI:266,SCTS:275,SPIPC T:784,SOC:340,TT:512,TTT:012,UNT:382,RV:082,HFUMain:554,RBFT:772,RT:205, VV:168,Pmain:129,Psub:031,SO:343,SOsub:625).

After the calculation of the average number X and deviation standard S, the lower bound of support 0.01 is obtained. If the Process' support is lower than 0.01, it is

considered as the less-occurred Process in all Processes. The result is displayed as follows: TTT, Psub, RPI, RV, RCFT and Pmain. Although data volume is insufficient, by means of the use of Random Numbers to simulate the execution frequency, it is feasible to validate the algorithm's accuracy.

Experiment 3- RedundantWM

Experimental Object: Mine the Process Activity sequence which is never executed. Find every Process' possible execution sequence in advance. Then, follow the procedure of section 3.3 to find Redundant Workflow. Besides, only nine Process' possible execution sequences are required to be found in this experiment because other twenty Processes are never-occurred.

Before reading data into the system is conducted, a special adjustment for Activity TR14 of Process RT_06 (Fig.10) needs to be made. Activity TR14's ACTIVITYTYPE is converted into SPLIT. SPLIT_NEXT is RT13 and RT8. The reason for this adjustment is that the path from RT7 to RT 13 not only is (RT7,8,9,10,11,12,13), (RT7,8,9,12,13) and (RT7,14,13), but also has the parallel path (RT7, 14,8,9,10,11,12,13) and (RT7,14,8,9,12,13).

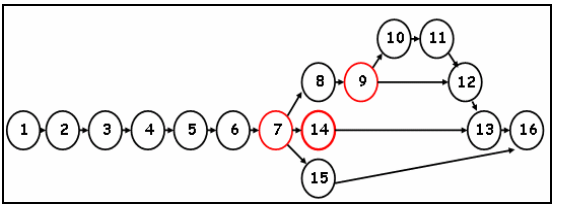

Fig. 10. RT_06 Process Model

The result of Process HFUMain_02, PPC_01, Pmain_04 is demonstrated in Fig.11, while the result of RT_06 and VV_01, Process Psub_12, RBFT_02 and SO_04 is similar.

The left side of figures is every Process' information, while the right side is the result of running algorithm. Set (all) represents sequences of all possible Process Activities. Instance(Pi) is ever-executed Workflow. RedundantWF stands for all the Workflows which are never-occurred, and null indicates zero existence of never-executed Workflow.

Fig.12 shows the result of Process SOSub_190, which has 190 possible paths. Some Activity SOS2 is presented as SOS2(30), which means thirty repeated execution paths appear repeatedly. According to Process SOSub_190's Process Model, there are sixteen possible paths. However, the algorithm only takes one loop iteration into consideration. If the execution path needs to be repeated more then once, there are only 30(15+15) ways to finish the Process. RedundantWF represents all the Workflows which never-occurred.

Experimental Analysis: In each Process' DEFINITIONID, the number indicates how many possible paths in each Process are calculated in advance; for example, SOSub_190 means Process SOSub has 190 possible execution paths. Set(all) stands for all the possible execution paths calculated by the algorithm.

DEFINITIONID	ACTIVITYNAME	ACTIVITYTYPE	SPLIT_NEXT	Set(all)	Instance(Pi)	RedundantWF
HFUMain_02	HFU1	START		(HFU1,2,3,4)	(HFU1,2,3,4)	(HFU1,2,3,2,3,4)
HFUMain_02	HFU2			(HFU1,2,3,2,3,4)		
HFUMain_02	HFU3	SPLIT	HFU2、HFU4			
HFUMain_02	HFU4	END				
DEFINITIONID	ACTIVITYNAME	ACTIVITYTYPE	SPLIT_NEXT	Set(all)	Instance(Pi)	RedundantWF
PPC_01	PPC1	START		(PPC1,2,3,4)	(PPC1,2,3,4)	null
PPC_01	PPC2					
PPC_01	PPC3					
PPC_01	PPC4	END				
DEFINITIONID	ACTIVITYNAME	ACTIVITYTYPE	SPLIT_NEXT	Set(all)	Instance(Pi)	RedundantWF
Pmain_04	PMA1	START		(PMA1,2,3,4,8,10)	(PMA1,2,3,4,8,10)	(PMA1,2,3,4,9,7,10)
Pmain_04	PMA2	SPLIT	PMA3、PMA5	(PMA1,2,3,4,9,7,10)		(PMA1,2,5,6,4,8,10)
Pmain_04	PMA3			(PMA1,2,5,6,4,8,10)		(PMA1,2,5,6,4,9,7,10)
Pmain_04	PMA4	SPLIT	PMA8、PMA9	(PMA1,2,5,6,4,9,7,10)		
Pmain_04	PMA5					
Pmain_04	PMA6					
Pmain_04	PMA7					
Pmain_04	PMA8					
Pmain_04	PMA9					
Pmain_04	PMA10	END				

Fig. 11. The Result of RedundantWM (1)

DEFINITIONID	ACTIVITYNAME	ACTIVITYTYPE	SPLIT_NEXT	Set(all)	Instance(Pi)	RedundantWF
SOSub_190	SOS1	START		(SOS1,2,3,4,5,6,7,17)	(SOS1,2,3,4,5,17)	(SOS1,2,3,4,5,6,7,17)
SOSub_190	SOS2	SPLIT	SOS3、SOS14	(SOS1,2,3,4,5,17)	(SOS1,2,3,8,4,5,17)	(SOS1,2,3,8,4,5,6,7,17)
SOSub_190	SOS3	SPLIT	SOS4、SOS8、SOS11	(SOS1,2,3,8,4,5,6,7,17)	(SOS1,2,3,8,9,10,11,12,13,17)	(SOS1,2,3,8,9,10,11,12,2[27])
SOSub_190	SOS4			(SOS1,2,3,8,4,5,17)	(SOS1,2,3,8,9,10,11,12,11,12,13,17)	(SOS1,2,3,8,9,10,11,12,11,12,2[30])
SOSub_190	SOS5	SPLIT	SOS6、SOS17	(SOS1,2,3,8,9,10,11,12,13,17)	(SOS1,2,3,8,9,10,11,12,2,3,8,9,10,11,12,13,17)	(SOS1,2,3,11,12,13,17)
SOSub_190	SOS6			(SOS1,2,3,8,9,10,11,12,11,12,13,17)	(SOS1,2,3,8,9,10,11,12,2,3,4,5,17)	(SOS1,2,3,11,12,2[30])
SOSub_190	SOS7			(SOS1,2,3,8,9,10,11,12,2[30])	(SOS1,2,3,8,9,10,11,12,2,3,8,4,5,17)	(SOS1,2,3,11,12,11,12,2[29])
SOSub_190	SOS8	SPLIT	SOS4、SOS9	(SOS1,2,3,8,9,10,11,12,2,3,4,5,17)	(SOS1,2,3,11,12,11,12,13,17)	(SOS1,2,3,11,12,11,12,2,3,4,5,6,7,17)
SOSub_190	SOS9			(SOS1,2,3,8,9,10,11,12,2,3,8,4,5,17)		(SOS1,2,14,15,16,11,12,13,17)
SOSub_190	SOS10			(SOS1,2,3,8,9,10,11,12,11,12,2[30])		(SOS1,2,14,15,16,11,12,11,12,13,17)
SOSub_190	SOS11			(SOS1,2,3,11,12,13,17)		(SOS1,2,14,15,16,11,12,2[30])
SOSub_190	SOS12	SPLIT	SOS13、SOS11、SOS2	(SOS1,2,3,11,12,11,12,13,17)		(SOS1,2,14,15,16,11,12,11,12,2[30])
SOSub_190	SOS13			(SOS1,2,3,11,12,2[30])		
SOSub_190	SOS14			(SOS1,2,3,11,12,11,12,2[30])		
SOSub_190	SOS15			(SOS1,2,14,15,16,11,12,13,17)		
SOSub_190	SOS16			(SOS1,2,14,15,16,11,12,11,12,13,17)		
SOSub_190	SOS17	END		(SOS1,2,14,15,16,11,12,2[30])		
				(SOS1,2,14,15,16,11,12,11,12,2[30])		

Fig. 12. The Result of RedundantWM (3)

5 Conclusions and Suggestions

The research proposes an algorithm which makes use of the workflow's executed frequency, the concept of distance-based outlier detection, empirical rules and Method of Exhaustion to mine three types of workflow outliers, including less-occurring workflow outliers of each process, less-occurring workflow outliers of all processes and never-occurring workflow outliers. Besides, this research also adopts real data to evaluate workflow mining feasibility. To sum up, these algorithms can help consultants and managers find workflow outliers and adjust them in anticipation of efficient execution of business processes.

Among three experiments of this research, we suffer from insufficient data volume. Here, suggestions from experiments are proffered below :(1)in order to gain any result worthy of being experimented, a data set that lasts for a long period is required, and (2) attempt to make the algorithm as impeccable and tenable as possible when facing every kind of Process.

In the midst of plentiful methods available to mine outliers, frequency-based and distance-based concepts are applied in this research by virtue of the features of the workflow. In the future, it is likely that other methods will be employed to develop the algorithm for mining workflow outlier. With regards to many the different methods for mining that have been developed, available for us to conclude which method can be conducted in each situation.

In addition, loop iteration is also a critical issue to deal with. In this research, the numbers of iterations of loop are reduced to one, but human judgment is involved. In future research, it is essential to integrate this issue into the algorithm, with a view to reducing any potential inaccuracy.

References

[1] Weijters, A.J.M.M., van der Aalst, W.M.P.: Process mining: discovering workflow models from event-based data. In: Kr€ose, B., de Rijke, M., Schreiber, G., van Someren, M. (eds.) Proceedings of the 13th Belgium–Netherlands Conference on Artificial Intelligence (BNAIC 2001), pp. 283–290 (2001)

[2] Weijters, A.J.M.M., van der Aalst, W.M.P.: Rediscovering workflow models from event-based data. In: Hoste, V., de Pauw, G. (eds.) Proceedings of the 11th Dutch-Belgian Conference on Machine Learning (Benelearn 2001), pp. 93–100 (2001)

[3] Arning, A., Agrawal, R., Raghavan, P.: A linear method for deviation detection in large databases. In: Proceedings of the KDD 1996, pp. 164–169 (1996)

[4] Breunig, M.M., Kriegel, H.P., Ng, R.T., Sander, J.: LOF: identifying density-based local outliers. In: Proceedings of the SIGMOD 2000, pp. 93–104 (2000)

[5] Schimm, G.: Process Mining, http://www.processmining.de/

[6] Gartner Group, http://www.comwave.com.tw/crm-solution/defi.htm

[7] Hawkins, D.: Identification of outliers. Chapman & Hall, Reading (1980)

[8] Herbst, J.: A machine learning approach to workflow management. In: Lopez de Mantaras, R., Plaza, E. (eds.) ECML 2000. LNCS (LNAI), vol. 1810, pp. 183–194. Springer, Heidelberg (2000)

[9] Herbst, J., Karagiannis, D.: An inductive approach to the acquisition and adaptation of workflow models. In: Ibrahim, M., Drabble, B. (eds.) Proceedings of the IJCAI 1999 Workshop on Intelligent Workflow and Process Management: The New Frontier for AI in Business, Stockholm, Sweden, pp. 52–57 (August 1999)
[10] Herbst, J., Karagiannis, D.: Integrating machine learning and workflow management to support acquisition and adaptation of workflow models. In: Proceedings of the Ninth International Workshop on Database and Expert Systems Applications, pp. 745–752. IEEE, Los Alamitos (1998)
[11] Herbst, J.: Dealing with concurrency in workflow induction. In: Baake, U., Zobel, R., Al-Akaidi, M. (eds.) European Concurrent Engineering Conference, SCS Europe (2000)
[12] Herbst, J.: Ein induktiver Ansatz zur Akquisition und Adaption von Workflow-Modellen, Ph.D. thesis, Universit€at Ulm (November 2001)
[13] Cook, J.E., Wolf, A.L.: Event-based detection of concurrency. In: Proceedings of the Sixth International Symposium on the Foundations of Software Engineering (FSE-6), pp. 35–45 (1998)
[14] Cook, J.E., Wolf, A.L.: Software process validation: Quantitatively measuring the correspondence of a process to a model. ACM Transactions on Software Engineering and Methodology 8(2), 147–176 (1999)
[15] Knorr, E., Ng, R.: A unified notion of outliers: Properties and computation. In: Proceedings of the KDD 1997, pp. 219–222 (1997)
[16] Knorr, E., Ng, R.: Algorithms for mining distance-based outliers in large datasets. In: Proceedings of the VLDB 1998, pp. 392–403 (1998)
[17] Knorr, E., Ng, R.: Finding intentional knowledge of distance-based outliers. In: Proceedings of the VLDB 1999, pp. 211–222 (1999)
[18] Jansen-Vullers, M.H., van der Aalst, W.M.P., Rosemann, M.: Mining configurable enterprise information systems. Data & Knowledge Engineering 56, 195–244 (2006)
[19] Agrawal, R., Gunopulos, D., Leymann, F.: Mining Process Models from Workflow Logs. In: Sixth International Conference on Extending Database Technology, pp. 469–483 (1998)
[20] Smith, H., Fingar, P.: Business Process Management: The Third Wave. Meghan-Kiffer Press, Tampa (2002)
[21] Yamanishi, K., Takeuchi, J.: A unifying framework for detecting outliers and change points from non-stationary time series data. In: KDD 2002, pp. 676–681 (2002)
[22] Yamanishi, K., Takeuchi, J.: Discovering outlier filtering rules from unlabeled data-combining a supervised learner with an unsupervised learner. In: Proceedings of the KDD 2001, pp. 389–394 (2001)
[23] Yamanishi, K., Takeuchi, J., Williams, G., Milne, P.: On-Line Unsupervised Outlier Detection Using Finite Mixtures with Discounting Learning Algorithms. Data Mining and Knowledge Discovery, 275–300 (2004)
[24] He, Z., Xu, X., Huang, J.Z., Deng, S.: Mining class outliers: concepts, algorithms and applications in CRM