A Method to Tackle Abnormal Event Logs Based on Process Mining

Zhanmin Yang[1]
[1]School of Computer Science and Technology
Shandong University
Jinan, China
zhm@sdu.edu.cn

Liqun Zhang[2], Yuan Hu[3]
[2]School of Computer Science and Technology
Shandong University
Jinan, China
zhanglq120@126.com

***Abstract*—One important function of process mining is to manage business process in order to detect the abnormal. A new process mining method is proposed based on process mining technology. The method will form a new process model that is more extensible relative to original process model. Then we use a effective method re-executing our event logs based on this new process model. Several kinds of abnormal are defined: task lost; task is excess; task replaced; tasks disordered. Due to the define of abnormal type, the re-executed results will show us the concrete mistakes about abnormal behaviors and help us understood and analysis the business process easily.**

Keywords—Process Mining, Abnormal Event Logs; Petri-net, Re-execute

I. INTRODUCTION

Nowadays, most of the companies or organizations use information-system to manage their business. There are several information-systems which support the process manage, for instance WFMS, CRM, ERP, etc. The usual cases executed in their departments are recorded as the logs which generally include the tasks in each case; start time or end time of each task; the managers who perform in each case; and other information about each case[1].These logs are the starting point of our following work, and are usually called event logs. In summary, the definition of process mining is: suppose that the real execution order of real cases are recorded by event logs, due to which we can rebuild a work-flow model, and make every case in the log fit one track of the model. Thus we can get a vivid show of how the real cases executed in the log[2][3][4]. This is one important function of the process mining, furthermore we should mine more information from the logs itself. The model should not only be used as a view, but also a standard to identify abnormal cases. Due to the workers' fault, it is hard to avoid some mistakes in our daily work. Therefore, it is necessary to mine a specialized process on the basis of daily work, and then build a simple model about the process to detect the event logs recorded from the process with the pre-defined mistakes. Thus the detected results will help the managers discover the abnormal event logs and the problems during the real process[5][6][7][15][16].

II. RELATED WORK

In fact the idea of deducing the process model from logs has been utilized. In 1995, Pro. Jonathan. E.Cook firstly put forward the idea in software engineering area[2]. In 1998, Rakesh Agrawal firstly applied the idea in workflow modeling area[8]. However, until recent years, researchers have been paid more attention to this modeling technology, and lead to a new modeling research direction.

Gudio Schimm has developed a mining tool (Process Miner) suitable for discovering hierarchically structured workflow processes [9][10]. This algorithm requires all splits and joins to be balanced. The main problem of this algorithm is difficult and impossible to let the discovered model cover numerous instances (e.g. parallel paths; loops).

Joachim Herbst and Dimitris Karagiann applied inductive approach and machine learning method in workflow mining[11][12]. The mining process is divided into two steps, which include induction step and transformation step. However the algorithm only deduced workflow models with sequential structure.

W.M.P van der Aalst and his research team did plenty of work in workflow mining[4][13]. They have designed several different workflow mining algorithms, which include formal algorithms (α , $\alpha+$, β) and heuristic algorithm. The mining results are described as WF-net. The limitation of the α algorithm is that cannot deal with the loops, invisible tasks and duplicate tasks.

W.M.P van der Aalst firstly put forward Workflow nets (WF-nets), creatively mapping workflow management concepts onto Pertri nets. The WF-net is the mainstream representation approach for modeling and analyzing workflow processes. Structured Workflow nets (SWF-nets) are a subclass of workflow nets (WF_nets) in which the structure explicitly shows its behavior[14].

A workflow process is designed to handle similar cases. Cases are handled by executing tasks in a specific order. The workflow process model specifies which tasks need to be executed and in what order. Tasks are modeled by transitions and causal dependencies are modeled by places and arcs. A workflow log is a sequence of events. We can consider a workflow log as a set of event sequences where each event sequence is simply a sequence of task identifiers.

Petri-NET: the classic Petri-net is a simple process model composed of two kinds of nodes (transition and place), directed arc, tokens and etc. The transaction nodes stand for the tasks in the event. If every input place of a transition has a token, it's enabled to fire (the corresponding task can be executed). Each of the input places of the transition will consume one token and each of the out places of the transition will produce one token after it fires. The relations between event tasks in the sound SWF-net response to event log W are defined as follows:

If and only if there is a path $\sigma = t_1t_2t_3\ldots t_{n-1}, i\in\{1,2,\ldots,n-2\}$,satisfy $\sigma\in W$ and $t_i=a$, $t_{i+1}=b$

$a\rightarrow_w b$ if and only if $a >_w b$ and $b \not> _w a$

a $\#_w$ b if and only if a $\not>_w$ b and b $\not>_w$ a

a $\|_w$ b if and only if a $>_w$ b and b $>_w$ a

WF-NET: N=(P,T,F) is a WF-NET(see Fig. 1) , where P is the set of places with tokens in them, T is the set of transitions, F is a set of directed arcs, called the flow relation. Places can only be connected with transitions, and transitions can only be connected with places. In the WF-net there are four relations[4][14] as follows:

Follow: the relations among T_0, P_0 and T_1. After T_0 executed, a token produced in P_0,and then T_1 can be executed.

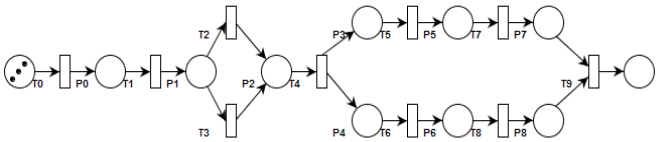

Fig. 1 Example of WF-NET

1) OR-SPLIT: the relations among P_1,T_2,T_3. Only one of T_2 and T_3 can be executed after consuming one token in P_1.

2) OR-JOIN: the relations among T_2,T_3,P_2. After one of T_2 and T_3 executed, a token produced in P_2.

3) AND-SPLIT: the relations among T_4,P_3,P_4. After T_4 executed, both P_3 and P_4 will increase one token and then T_5,T_6 can be executed.

4) AND-JOIN: the relations among P_7,P_8,T_9. When both of P_7,P_8 have a token, T_9 can be executed, and then the token in both P_7,P_8 consumed.

III. BUILDING MODEL

Now let's have a general view of our model, see Fig. 2.

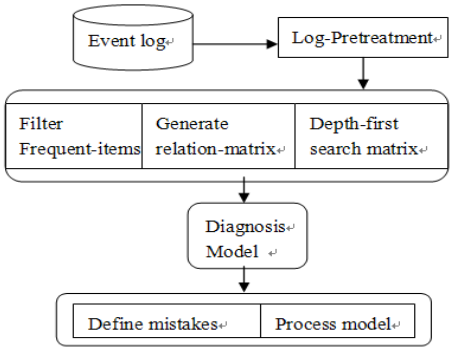

Fig. 2.Our building model.

First of all format event log to xml document, in order to read convenient. Each document has some cases, each case has some tasks, and each task has some attribute nodes, such as task name;task attributes; task's start time; end time;executer; resources. We need to mine a Petri-net from these information in the document. Due to our algorithm's limitation, only the logs which can form SWF-net[14] will be considered.

Now let's begin our mining process:

A.Set threshold. This threshold is used to remove some noise data which appear less than it[6].

B.Traverses the whole logs. Find out all the tasks appeared in the logs and the times they appear. Record them as 1-item, then reduce the 1-items which less than the threshold, and remove all the cases contain them.

C.Third, mining out pairs of task due to their relations, and record the times they occur. Traverses all the cases until no more new pair appears (Fig. 3, Fig. 4). Set the threshold 2.Pairs EG and GF only occur once, remove them.

Case 1	ABCDEF
Case 2	ACBDEF
Case 3	ACBDEF
Case 4	ABCDEF
Case 5	ABCDEGF

Fig. 3.Some cases.

AB	2
BC	2
CD	2
…	…
CB	2
(EG)	1
(GF)	1
…	……

Fig. 4.Pairs of tasks

	A	B	C	D
A	0	1	1	0
B	0	0	1	1
C	0	1	0	1
D	0	0	0	0

Fig. 5.The relation matrix.

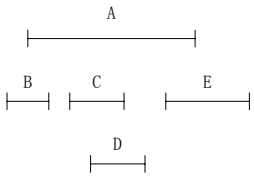

Fig. 6. Parallel cases

There are two situations should be added.

One is: When the task in the case has both start and end records. The parallel relation should be described as "one task starts before another task ends then they are parallel" [4][6].In Fig. 6,AB,AC,AE,CD,AD are paralleled[4]. Change the record method of task. If a task A has both a start and a end records. It's recorded as A(start) and A(complete)(As, Ae for short). Simply we need not record the relation between As and Ae. Define two relations between task A and B as follow:

a) If exits AeBs, in the corresponding matrix Ae row Bs line is recorded as 1, the relation between AB is A→ B;

b) If exits AsBsAeBe or AsBsBeAe, in the corresponding matrix AsBs, BsAe, AeBe are all 1 or AsBs,BeAe are 1. the relation between AB is A||B;

Second is : local non-free choice, if task C and its followed task have a OR-SPLIT relation, and the choice is decided by C's attribute. Then we should consider that bring in C's attributes to our cases. Such as adding CC_1,CC_2(C_1,C_2 are two different attributes in C, each can lead to a different following task).

D. Construct a matrix based the pairs produced in the third step. If there is a pair of AB then the value AB in the matrix is 1, else is 0(Fig. 5).

E. Mine from the matrix. Our algorithm is described as follow:

First of all define two kinds of nodes[3][4], transaction and place.(The tasks in the cases response to transaction nodes)

```
class T
{                       //transactions
    P[] prePlaces;  //places direct to this transaction
    P[] nextPlaces;     //places be directed by this
                        // transaction
    List Attributes;   //attributes of this transaction
    String Name;     //transaction name;
    boolean flag;
```

```
    }
class P
{                       //places
    T[] preTransactions;//transactions direct to this
                        //place
    T[] nextTransactions;//transactions be directed
                         // by this place
    List Attributes;   //attributes of this transaction
    String Name;       //transaction name
    int token;         //token left
    int inTokens;      //tokens produced by
                       //preTransactions
    int outTokens;     //tokens consumed by
                       //nextTransactions
    }
```

Our algorithm adopts a depth first search strategy to traverse the matrix to form a special tree; of course it's a Petri-net. In the tree, there maybe have some circles, the nodes of the tree are composed of places and transactions. The algorithm starts with a transaction, find the next transaction from the matrix until encounters an existed transaction or an end transaction. When encounters an existed transaction it will start with its other inexistent brother node again and go ahead.[7]

```
GeneratePetriNet(int[][] M,T s, List E,Net net)
{
/*M is the relation matrix,s is the start transaction, E is the set
of end transactions.net is the Petri-net
we wanted.*/
   if(s is not in E){
     if(all transactions have been visited)
        return;
     NetList.add(s);   //NetList is the set of
                       //transactions have been visited
     List nodeList = getNodes(s);
/* get all the follow nodes of s according to the matrix.The
value in matrix is 1 in row s.*/
    for(each node x in nodeList){
   if(check(x,NetList))          /*if x is already in NetList,we
should consider the relation between s and x's preNodes. The
value in matrix is 1 in line x.*/

   {……/*This process is similar to following. The relation of
them will be OR-JOIN or AND-JOIN.*/

    remove x from nodeList;}}
    //thus no node in nodeList has been visited;
    for(each node I in nodeList){
       net.add(I);       //add I to net;
     }
boolean rel = checkRelations(nodeList);
    /*return the relation among nodes in nodeList, OR-SPLIT
or  AND-SPLIT*/
if(rel){          //OR-SPLIT;
  List orList = getOrNodes(nodeList);/*orList contains N items
of or-relation.nodes in each item are parallel*/

    if(every item in orList have only one node){
       newPlace (flag,net);/*create a new place in the net,flag is
used to distinguish places*/
       newLine(s, flag,net);/* create the relation between place
and transaction.and flag++ */
       for(every node x in each item of orList)
          newLine(flag, x,net);
       } flag++;}
   else{         //some items have more than one nodes;
          for(every node y in each items which have more than
one nodes)
            {newPlace (flag,net);
              newLine(s, flag,net);
              newLine(flag,y,net);
              for(all the nodes in orList which have no parallel-
relation with y)
                 newLine(flag ,z,net);
              }
         flag++;}
else{ // AND-SPLIT
     List andList = getAndNodes(nodeList);
     for(each item k in andList){//nodes in k are or-relation
        newPlace(flag ,net);
        newLine(s, flag ,net);
        for(each node in k)
           newLine(flag, k ,net);
     }
     flag++;
  }
  for(each node z in nodeList){ // recursive calls
     GeneratePetriNet(M, z ,E,net);
     }
  }
```

F. According to net and the places, transactions produced in the process, we transform them to XML DOM form. With the help of Prom tools, we will finally get our model.

To test and verify our method, we get some event logs from a mobile-repair department. Using the process above, we get the visual model. (Fig. 7)

IV. PROCESS DIAGNOSIS

After the process model is built, we should build our diagnosis model next. So we choose the re-execute method to deal with event-logs and Petri-net.

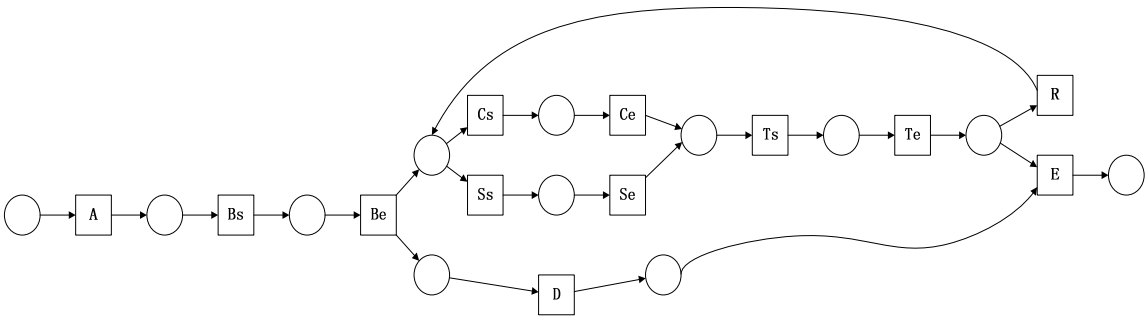

Fig. 7.An example of the model based on our building model

First of all, let's have a general view of the method. Assume that all the transaction (in a normal case, if there is OR-Split only one path should be selected) can be fired from the beginning transaction. After all firing, every of its prePlaces reduces one token(outTokens-1), every of its nextPlaces adds one token. Some places' tokens are not 0, if the places' tokens >0 then the inTokens +1 else the outTokens -1. The inTokens and outTokens of these places are the basis of our diagnosis.

In order to analysis abnormal behavior conveniently, here we define several kinds of mistakes. a: task lost; b: task is excess; c: task replaced; d: tasks disordered. inTokens are marked with "+", outTokens are marked with "-".

For example there are four cases a, b, c, d.

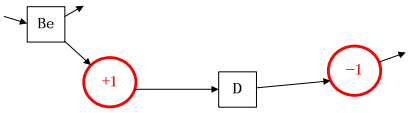

Fig. 7-1. The result of case a.

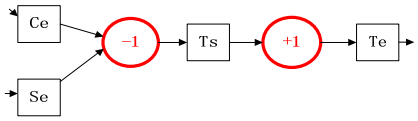

Fig. 7-2.The result of case b.

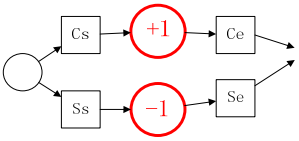

Fig. 7-3.The result of case c.

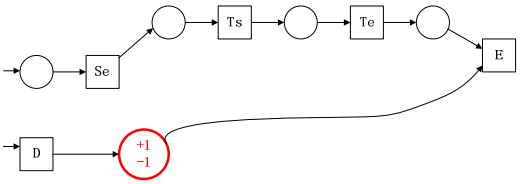

Fig. 7-4.The result of case d.

a: A Bs Be Cs Ce Ts Te E

We deal it as the input of process model (Fig. 7).After transaction Be executed, the place between Be and D produces one token, but transaction D doesn't executed. Thus the token of the place between Be and D is 1(inTokens is +1), the token of the place between D and E is -1(outTokens is -1).Be need a next-transaction and E need a pre-transaction. D is lost(Fig. 7-1).

b: A Bs Be Cs Ce D Ts Te Ts E

Shown in Fig. 7-2.The second Ts'prePlaces has one outTokens(-1) and Ts'nextPlaces has one inTokens(+1), so this Ts has no pre-transaction, it's excrescent.

c: A Bs Be Cs Se D Ts Te E

Shown in Fig. 7-3.There are two mistakes. Cs need a next-transaction and Se need a pre-transaction. Likely, Ce is replaced by Se or Ss is replaced by Cs.

d: A Bs Be Cs Ce Ts Te E D

Shown in Fig. 7-4.Transaction E need a pre-transaction, but its pre-transaction D in the model is existed and D also need a next-transaction. Thus E or D probablely have a disordered relation.

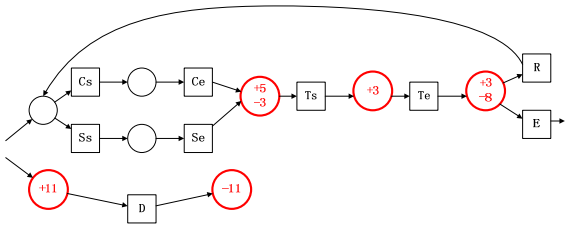

Fig. 7-5. The result of real events

Traverse all the cases(1000 cases), the final result is shown in Fig. 7-5.There may be 11 D lost,3 T(Ts,Te) excrescent, several R position mistakes. Thus we can inform that transaction D,T and R are often made mistakes.

V. CONCLUSION

In this paper, we summarized a kind of diagnosis model based on process model. The process model is built from a depth first search strategy, the relations between nodes we defined are similar with α-algorithm. In our process, it's easy to extend, such as change task's attributes to tasks or define relation of different parallel tasks. Our diagnosis model uses the Re-execute Petri-net to traverse the event logs based the process model. The result clearly show that the abnormal cases are tackled and evenly show us the concrete mistakes which is the most advantage over other diagnosis model [15][17][18]. The shortcoming of the model is that the model is built on

SWF-net, we can't mine out some special structures but common in practice[14].

REFERENCES

[1] Hou Zhisong, Yu Zhou, Research of the workflow management system based on microkernel. *Journal of Theoretical and Applied Information Technology,* 2013, Vol.47, No.1, pp.266-271.

[2] Van Der Aalst, Wil M.P. and Dustdar, Schahram, Process mining put into context, *IEEE Internet Computing*, Vol. 16, pp. 82-86, 2012.

[3] R. Agrawal, D. Gunopulos, and F. Leymann, Mining Process Models from Workflow Logs. *In Sixth International Conference on Extending Database Technology*, pp. 469–483, 1998.

[4] W.M.P. van der Aalst, A.J.M.M. Weijters, and L. Marust, Workflow mining: Discovering process models from event logs, *IEEE Transactions on Knowledge and Data Engineering*, Vol.16,No.9, pp.1128~1142 September 2004.

[5] W.M.P. van der Aalst, B.F. van Dongen, Discovering Workflow Performance Models from Timed Logs, *In Y. Han, S. Tai, and D. Wikarski, editors, International Conference on Engineering and Deployment of Cooperative Information Systems (EDCIS 2002)*, Vol.2480 of Lecture Notes in Computer Science, pp. 45–63. Springer-Verlag, Berlin, 2002.

[6] Melike Bozkaya, Joost Gabriels, Process Diagnostics: A Method Based on Process Mining. *Information, Process, and Knowledge Management, 2009. eKNOW 09. International Conference* pp. 22 - 27 , Cancun,Feb. 2009.

[7] Liqun Zhang, Research on Block Structured Process Mining Techonology for Business Process Modeling. *Ph.D. Thesis, Shandong University, Jinan Shandong, P.R.China*. April 5,2010.

[8] R. Agrawal, D. Gunopulos, F. Leymann, Mining process models from workflow logs. *In Sixth International Conference on Extending Database Technology.* pp.469–483,1998

[9] G. Schimm, Generic Linear Business Process Modeling. *Proceedings of the ER2000 Workshop on Conceptual Approaches for E-Business and The World Wide Web and Conceptual Modeling*. Springer-Verlag, Berlin, 2000, Vol.1921 of LNCS,pp.31–39.

[10] G. Schimm, Process Miner-A Tool for Mining Process Schemes from Event-based Data. *Proceedings of the 8th European Conference on Artificial Intelligence (JELIA)*. Springer-Verlag, Berlin, 2002.

[11] J. Herbst, D. Karagiannis, Integrating machine learning and workflow management to support acquisition and adaptation of workflow models. *International Journal of Intelligent Systems in Accounting, Finance and Management.* pp. 67–92. 2000,9.

[12] J. Herbst, Dealing with concurrency in workflow induction. *Concurrent Engineering Conference. Society of Computer Simulation (SCS)*. Europe, 2000

[13] A.K.A. de Medeiros, W.M.P. van der Aalst, and A.J.M.M. Weijters, Workflow Mining: Current Status and Future Directions, *On The Move to Meaningful Internet Systems 2003.* Springer-Verlag, Berlin, 2003, Vol.2888 of Lecture Notes in Computer Science,pp. 389-406.

[14] Wil van der Aalst, Kees van Hee, WorkFlow management: models, methods and systems. *MIT Press*.2004.3.

[15] W.-S. Yang, S.-Y. Hwang, A Process-Mining Framework for the Detection of Healthcare Fraud and Abuse. *Expert Systems with Applications*, 2006,31() pp.56-68.

[16] Kartit A., Saidi A., Bezzazi F., El Marraki, M. Radi A. A new approach to intrusion detection system. *Journal of Theoretical and Applied Information Technology,* 2012, Vol.36,No.2, pp.2846-289.

[17] Gun-Woo Kim, Seung Hoon Lee, Jae Hyung Kim, Jin HyunSon, An Effective Algorithm for Business Process Mining Based on Modified FP-Tree Algorithm. *Communication Software and Networks, 2010. ICCSN '10. Second International Conference* pp.119 - 123, Feb. 2010.

[18]. Luengo, Daniela, and Sepulveda, Applying clustering in process mining to find different versions of a business process that changes over time. *Lecture Notes in Business Information Processing*, Vol.99 LNBIP, PART 1, pp. 153-158, 2012.