Rationality Validation of Business Process Model by Simulation Method

Ke Ning[1], Yuliu Chen[2], David O'Sullivan[1]

[1] Digital Enterprise Research Institute, National University of Ireland, Galway, Ireland
[2] Department of Automation, Tsinghua University, Beijing, 100084, China
Ke.ning@deri.org

ABSTRACT

Business process modeling has been adopted widely. Due to the complexity, it is very hard to validate the rationality of process models. The existing validation methods can only detect structural conflicts in process models. It is also very important to validate those objects related to the processes. This paper presents a rationality validation method which is based on discrete event simulation technology. It can detect three logic mistakes from business process models: structural deadlock, lack of synchronization and objects not matched each other. This method has extended the scope of rationality validation, and also enriches the contents that can be validated.

Keywords: Business Process Models, Rationality Validation, Simulation

1. INTRODUCTION

Business process model is kind of description of business process. By using model, we are able to analyze the performance of process, to guide the implementation of process, to monitor the execution of process, to enable the management of process. Due to the complexity of business process, whether describing AS-IS models or designing TO-BE models, it is possible to make kinds of mistakes about rationality. It is nonsense to analyze the performance of a process based on the wrong models, and implementing a wrong model may bring huge losing to an enterprise. Therefore, more and more attention has been pay to the validation of business process model.

Validation of a business process model is not so easy. If the structure and size are not limited, it will be an NP hard problem to validating a process model [1]. Thus many efforts have been pay to how to improve the efficiency of validation algorithm, hoping to get high efficiency through simplifying model or adding restrict conditions. H. Lin [2] gave a graphic reduction method to validate the mistakes of structural deadlock and lack of synchronization of a process based on his process modeling language. W.M.P. [3] defined a WF-NET based on Petri-Net, then used Petri-Net theories to validate the rationality of the structure of a process. These methods have gain efficiency of multinomial time complexity. But problems raised here: Firstly, there are rigorous conditions for the algorithms, the method of H. Lin can validate only some specified simple structure of process, method based on Petri-Net required that the process model must be a free-choice net. Secondly, these methods can only find some logic problem of process structure, which is only a little scope of process rationality validation. These methods may have theoretical values, but they are not the practical ones for the real business process.

To support validation of real business process, we have gone on a different way. We haven't cared too much about efficiency of algorithm, but have paid more attention to enlarge the scope of contents of validation, and to improve the practicability. We have developed a validation method based on simulation technology in our eIDEF3 business process model. This method can detect three logic mistakes in business process: structural deadlock, lack of synchronization and objects not matched each other. Compared to the existing methods, it is not so effective, but enough for most business process. And most important, it has extended the scope of rationality validation, and also enriches the contents that can be validated.

2. EIDEF3 PROCESS MODELING METHOD

IDEF3 is one of the IDEF series methods. It is used to capture and describe business process. It can support communication and comprehension between domain experts and modelers, and has been used widely in industry. To increase its description ability, support simulation analysis, enlarge model usability, we have extended IDEF3 to eIDEF3 (extended IDEF3), which can describe objects involved in process in a formal way. We have developed a software tool GEM-EASY IDEF3© based on this method. Figure 1 is the basic syntactical elements of eIDEF3.

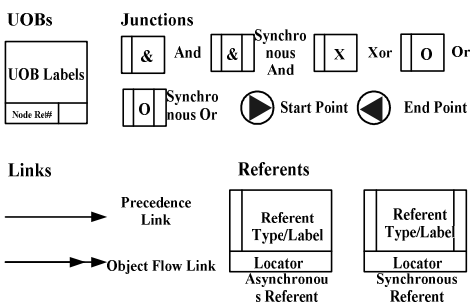
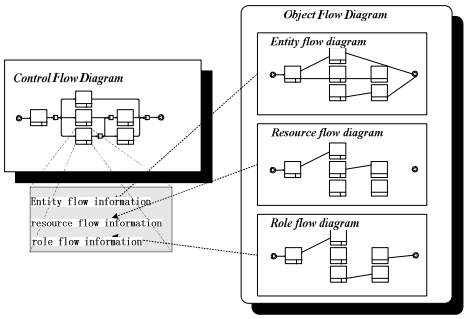
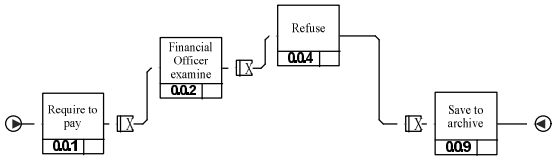


Figure 1 basic syntactical elements of eIDEF3

eIDEF3 provides two type basic flow diagram to describe a process scenario (As shown in Figure 2): One is Control Flow diagram, the other is Object Flow Diagram.





$$\begin{aligned} &\forall n \in N, \\ &if\ (n \neq BeginOf\ (pis) \wedge n \neq EndOf\ (pis)) \\ &then(n \notin PB \wedge n \notin PE) \end{aligned} \tag{5}$$

It is obviously that this definition of process nationality emphasizes the normally beginning and normally ending of a process. This is consistent with people's intuitive comprehension. More important, this definition is given from process itself, and is not bounded with specified modeling methods. So it is more generic. Moreover, defining process rationality based on PIS has built the foundation to validate it by using simulation method.

4. PROCESS RATIONALITY CRITERION IN EIDEF3 MODEL

Before give the concrete algorithm, we will give the criterion based on some precondition and assumption.

4.1Basic precondition and assumption

Based on the requirement of eIDEF3 syntactical rules, the following basic precondition must be satisfied.

1) An eIDEF3 model P is an all connected diagram from Start Point to End Point. No isolated nodes exist. That is:

$$\begin{aligned} &\forall n \in P, \\ &InOf\ (n) \geq 0 \vee OutOf\ (n) \geq 0 \end{aligned} \tag{6}$$

2) Connection rules: The Precedence Links of UOB are single in and single out. The Precedence Links of FanOut Junction are single in and multi out. The Precedence Links of FanIn Junction are multi in and single out. The Precedence Links of Start Point are none in and multi out. The Precedence Links of End Point are multi in and none out. That is:

$$\begin{aligned} &\forall n \in P, \\ &if\ (n \in UOB) then (InOf\ (n) \leq 1 \wedge OutOf\ (n) \leq 1) \\ &if\ (n \in FanOut\_Juction) then (InOf\ (n) \leq 1) \\ &if\ (n \in FanIn\_Juction) then (OutOf\ (n) \leq 1) \\ &if\ (n = Start\_Point) then (InOf\ (n) = 0) \\ &f\ (n = End\_Point) then (OutOf\ (n) = 0) \end{aligned} \tag{7}$$

Moreover, we give some basic assumption here, which is only for facilitating description of validation problems, and will not influence the applicability of the algorithm:

1) Synchronous and Asynchronous Junctions are not differentiated. Alternate from synchronous to asynchronous, or from asynchronous to synchronous, will not change the Process Rationality.

2) There is no "Or" junction in process models. Based on the semantic of the "Or" junction, we can use a combination of "And" and "Xor" junction to replace it.

3) There is no feedback in a process. We assume that a process is a DAG （Directed Acyclic Graph). If a feedback is needed, we can put the feedback parts into the decomposed scenario, so that we can avoid dealing with it when using the validation algorithm. Then the rationality of the feedback parts can be determined by other methods.

4.2 Criterion of Process Rationality

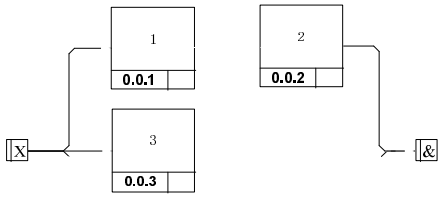
We have provided three criterions to validate process rationality, below are the details:

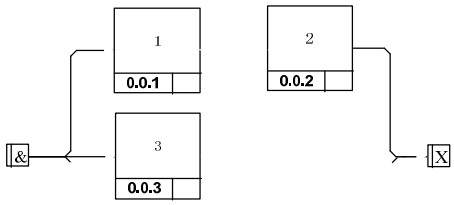
1) Structural Deadlock，SD

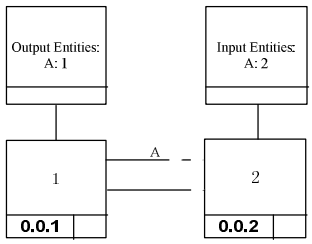
If two or more of the FanOut branchs of "FanOut Xor" junctions joint in a " FanIn And" junction, SD will happen. That is:

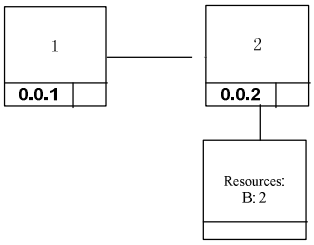
$$\begin{aligned} &\exists n \in FanOut\_Xor\_Jucntion, \\ &\quad OutOf\ (n) \geq 2 \wedge \\ &\quad \exists n' \in FanIn\_And\_Jucntion, \\ &\qquad n' \in OutPath(n)_i \wedge n' \in OutPath(n)_j, where\ i \neq j \\ &\Leftrightarrow P\ is\ SD \end{aligned} \tag{8}$$

As shown in figure 4, in this situation, the "FanIn And" junction will not be able to be triggered because it can not finsh all of its FanIn branch. Thus the process can not execute any more.




1
0.0.1
2
0.0.2
3
0.0.3
&
X


Output Entities:
A: 1
Input Entities:
A: 2
A
1
0.0.1
2
0.0.2


1
0.0.1
2
0.0.2
Resources:
B: 2

ELSE
{
Get a rational pis;
}
4. After getting a rational pis, select randomly one of the rest FanOut branck form the tailender "FanOut Xor" junction, repeat step 3, create a new pis;
5. Repeat step 4, until all the "FanOut Xor" junction have been deal with. Then we get all the PISs of the process. Then we can conclude that this process is a rational one;
END

What have to point out is, this algorithm can find the mistake of rationality, and find the node where the mistake exists. But it can not correct the mistake automatically. You have to correct it manually, and execute the algorithm again, because new change may bring new mistakes in other place. Until no mistakes are found, you get a rational process finally.

5.2 Analysis of the algorithm

1) Completeness
From the step 4 and step 5, we can see that for a rational process, this algorithm can deal with all the fanout branch of the "FanOut Xor" junction. So we can say that this algorithm can explore all PISs of the process.
Once there is mistake in the process, the algorithm will come to step 3. Here it can tell where the mistake is. Users can use this message to correct models, and run the algorithm again. Until all mistakes have been corrected, the process will become rational. According to the analysis of last paragraph, all the PISs of a rational process can be explored completely.
Therefore, this algorithm is completed. It can explore all PISs of a process, and find all mistakes in it.

2) Complexity
It is easy to conclude that for a rational process, to explore all the PISs, the time consumed will increase exponentially along with the amount of "FanOut Xor" junction:

$$O = \overline{c}_{path}^{\,c_{FanOut\_Xor\_Junction}} \tag{14}$$

Here, $\overline{c}_{path}$ is the average amount of fanout branch of the "FanOut Xor" junctions.

3) Usability
As we have mentioned at the beginning of this paper, most existing validation methods have many limitation in their usability. For example, the method of H. Lin can not validate complex models; the method of W.M.P requires the models must be free-choice. Our method has no rigorous limitation on process models, and the contents that can be validated are more abundant and complete.

Compared to those methods that are based on Petri-net, the complexity of our method is much better. For most enterprises, a business process will not include too many "FanOut Xor" junctions (where decision making is needed). In the Chinese Aviation CIMS Project, we had build lots of business process models, among these models, the most complex model had no more the 10 "FanOut Xor" junctions. For today's computer, it is very easy to deal with such a complexity.

5. CONCLUSION

This paper presents a simulation-based business process rationality validation method. It has extended the scope of rationality validation, and also enriched the contents that can be validated. Although it is not the most efficient one, but for most business processes it is enough. So it is a validation method that is very suited for business process validation. Although this method is created based on eIDEF3, but the concepts about process rationality, process instance sub-graph, are independent of concrete modeling methods. So it is very easy to extend this method to other models, e.g. Petri-net models, workflow models, etc.

ACKNOWLEDGEMENT

The main result of this paper was created during my doctroal research, which was supported by China 863 Program. The publication of this paper was supported by Scientific Fundation of Ireland.

REFERENCES

[1] Hofstede A.H.M., Orlowska M.E., Rajapakse J. Verification Problems in Comceptual Workflow Specifications. Proceedings of the 15th International Conference on Conceptual Modeling, volume 1157 of Lecture Notes in Computer Science. Cottbus, Germany: Springer-Verlag, pp73-88, 1995.
[2] H. Lin, Z. Zhao, H. Li, and Z. Chen. A Novel Graph Reduction Algorithm to Identify Structural Conflicts[A]. Proceedings of the Thirty-Fourth Annual Hawaii International Conference on System Science. IEEE Computer Society Press, 2002. 536, 540, 541, 546, 548, 550.
[3] W.M.P. vander Aslst, Arthur HM, Ter Hofstede. Verification of Workflow Task Structures: A Petri-Net-Based Approach. Information Systems, 2000, 25(1): 43〜69.
[4] NIU Dong, NING Ke, LI Qing, CHEN Yuliu. Method and Tool for IDEF3 Based Process Modeling[J]. Computer Integrated Manufacturing Systems, 2001, 7(12): 30-34.
[5] NING Ke, NIU Dong, LI Qing, CHEN Yuliu. IDEF3-based Business Process Simulation Modeling[J]. Computer Integrated Manufacturing Systems-CIMS, 2003, 9(5): 351-356.
[6] Aalst W.MP. Verification of Workflow Nets. Application and Theory of Petri Nets 1997, volume 1248 of Lecture Notes in Computer Science, Berlin: Springer-Verlag, 1997. 407~425.
[7] Wasim Sadiq, Maria E.Orlawska. Analyzing Process Models Using Graph Reduction Techniques. Information Systems, 2000, 25(2): 117〜134.