Simplified Process Model Discovery Based on Role-Oriented Genetic Mining

Zhao Weidong[1], Liu Xi[1], Weihui Dai[2]

[1]Software School, Fudan University, Shanghai

[2]School of Management, Fudan University, Shanghai

Correspondence should be addressed to Weihui Dai,whdai@fudan.edu.cn

Process mining is automated acquisition of process models from event logs. Although many process mining techniques have been developed, most of which mine process models based on control flow. Meanwhile, the existing role-oriented process mining methods focus on correctness and integrity of roles while ignoring role complexity of the process model, which directly impacts understandability and quality of the model. To address these problems, we propose a genetic programming approach to mine the simplified process model. Using a new metric of process complexity in terms of roles as the fitness function, we can find simpler process models. The new role complexity metric of process models is designed from role cohesion and coupling, and applied to discover roles in process models. Moreover, the higher fitness derived from role complexity metric also provides a guideline for redesigning process models. Finally, we conduct case study and experiments to show the proposed method is more effective for streamlining the process by comparing with related studies.

1. Introduction

Information systems are mostly driven by the process model. Therefore, the process model is a key factor for the system to run effectively. Process mining techniques aim to automatically generate process models by analyzing the event log, which assist the redesign of process models.

Process mining first appeared in the field of software engineering. It is proposed by Joanthan from New Mexico State University in 1995 [1]. Then, Agrawal started to apply process mining to process management in 1998 [2]. He used directed graphs to represent the association between different activities in business processes. Instead of using directed graph, Aalst used the workflow net, which is a subclass of Petri nets to represent process models. Based on the work, some scholars extended process mining algorithm to handle business logic, including sequence, parallel and circular relationship [3]. Compared with other process mining algorithms, genetic mining proposed by W. van der Aalst is a global search algorithm, dealing with noise effectively [4].

The structure of process models is often complex. They consist of circular, parallel, choice and hidden structures. Current process mining algorithms are not well developed in dealing with these structures. Process mining aims to find the mode from the process execution log which most closely matches the actual behavior of business processes. But with the complication of the process, there will be large amounts of alternative process models. How to find out the process model with low complexity is necessary

for process improvement. For example, Tian proposed a process mining algorithm combining genetic algorithms and simulated annealing algorithm [5]. Through analyzing process participants, the algorithm built a causal relationship matrix mapping process instances to the population chromosomes and mined the process model effectively. In fact, those methods are very complex, and the process model mined may be very complicated too. Then, some researchers employed the complexity metric of control flow, and computed the structural complexity of process models to guide process redesign [6]. However, these approaches only pay attention to the complexity of the process model from the perspective of control flow. How to discover process models with low complexity, especially from the organizational view and streamline the collaboration between process actors are necessary.

At present, most of process mining methods are based on process activities. They neglect the fact that the process depends on the collaboration between multiple roles. Though some scholars come to extract knowledge from the role perspective, their studies on the relationship between process roles are not complete and merely confined to discuss the interaction among organizational entities [7, 8]. Actually, the relationship between them is very complicated, so it is hard to uncover hidden information and the role complexity of business processes is ignored.

The remainder of the paper is organized as follows: section 2 introduces the role complex metrics. Then, the role complex fitness is shown in section 3 with the case study conducted in section 4. Moreover, we conduct comparative experiments in section 5. Finally, section 6 concludes the paper.

2. The role complexity of process models

The complexity of business processes describes process models from different perspective. It implies whether a business process model has right size, clear structure and is easy to understand and reasonably modular. Therefore, it is necessary to design process models with low complexity.

The previous studies focus on control flow that is composed of activities and their relationships. For example, Jorge discussed the complexity metric of control flow through experiments [9]. In addition, Vanderfeesten proposed cohesion and coupling metrics for process design [10]. However, a process is the integration of participants (roles), resources, objectives, information and business rules and so on. Control flow is just one of the factors affecting process complexity. More researchers begin to analyze business processes from different aspects.

2.1 The role cohesion metric

The role cohesion analyzes closeness of multiple activities performed by one role. It proposes that the activities performed by the same role have closer relationship. For example, if the activities performed by a role are based on the same data or require similar capacities, then the role may have greater role cohesion, and be more efficient to take the activities. The role cohesion metric is categorized into the following types.

1) The role activity cohesion is to assess the interaction between roles in terms of control flow. The shorter the interval is, the higher the role activity cohesion is. Herein, the interval between two activities is defined as the number of activities between them. For example, there are n activities between the activities a_1 and a_2. Then, we can define the distance between them as $L(a_1,a_2) = n+1$. Actually, there may be several execution sequences containing a_1 and a_2. Say that there are v execution sequences which contain a_1 and a_2, we can define the distance between a_1 and a_2 as

$$L(a_1, a_2) = \sum_{k=1}^{v} L_k(a_1, a_2) / v \quad (1)$$

We can examine the interval of every two activities by a role to measure the role activity cohesion. So the role activity cohesion of r can be defined as

$$\alpha_r = (1+ \sum_{a_1 \neq a_2 \& a_1, a_2 \in T(r)} (\max L - L(a_1, a_2)) / 2) / (C^2_{T(r)} \times \max L) \quad (2)$$

Where $T(r)$ represents all activities performed by r, and max L is the maximum distance between activities of the process separately, and $C^2_{T(r)}$ means the number of situations of activities' combinations. So, the role activity cohesion reflects the distance of activities performed by r. The shorter the distance is, the higher the role activity cohesion of r is.

2)The role data cohesion measures the cohesion between roles in terms of data. It analyzes the frequency of using different data.

Provided that $*a$ is the input data set of a, which is necessary for a, and $a*$ is the output data set of a. Then, $I = *a \cup a*$ is the data set, which is related to a, and $|I|$ is the number of elements in I. Then the role data cohesion of r is defined as

$$\beta_r = (1+ \sum_{a_1 \neq a_2 \& a_1, a_2 \in T(r)} |I_1 \cap I_2| / 2) / (\sum_{a_1 \neq a_2 \& a_1, a_2 \in T(r)} |I_1 \cup I_2| / 2) \quad (3)$$

The role data cohesion indicates the proportion the input and output data of activities by r have in common. The more they share the same data, the higher the role data cohesion of r is.

3)The role ability cohesion measures the cohesion between roles in terms of abilities needed. It computes what abilities are required for the role to perform different activities. If E is the set of abilities necessary to perform a, and $|E|$ is the number of elements in E, then the role ability cohesion of r is defined as

$$\chi_r = (1+ \sum_{a_1 \neq a_2 \& a_1, a_2 \in T(r)} |E_1 \cap E_2| / 2) / (\sum_{a_1 \neq a_2 \& a_1, a_2 \in T(r)} |E_1 \cup E_2| / 2) \quad (4)$$

The role ability cohesion shows how many kinds of abilities are required by activities performed by r have in common. The more they share, the higher the role ability cohesion of r is.

As a whole, the role cohesion metric is computed as followings:

$$Coh_r = \alpha_r \times \beta_r \times \chi_r \quad (5)$$

2.2 The role coupling metric

The role coupling metric implies the degree of association between activities taken by different roles. If there are several kinds of connections between activities performed by two roles, and one role is connected with more roles, it has greater role coupling metric.

1)The role activity coupling shows the degree of association between activities performed by different roles in a process. If activities by different roles are connected, these roles are interrelated. There are several kinds of connections corresponding to different degree of association.

Assume that r is responsible for a_1, and a_2 is not performed by role r. m and n represent the out-degree and in-degree of the connector between a_1 and a_2 separately. We can define the coupling weight as follows through the connection form between a_1 and a_2.

- If a_1 and a_2 are directly connected, then a_1 and a_2 are coupled, so the coupling weight between them is 1.
- If a_1 and a_2 are connected through AND connector, then a_1 and a_2 are also coupled, so the coupling weight between them is 1.
- If a_1 and a_2 are connected through OR connector, then the probability of coupling between them is

$\frac{2^{m-1} \times 2^{n-1}}{(2^m - 1) \times (2^n - 1)}$, so the coupling weight between them is $\frac{2^{m-1} \times 2^{n-1}}{(2^m - 1) \times (2^n - 1)}$.

- If a_1 and a_2 are connected through XOR connector, then the probability of coupling and coupling weight between them are both 1/mn.
- If a_1 and a_2 are not connected, they cannot be coupled, so the coupling weight between them is 0.

The role coupling metric of r is defined as

$$\delta_r = \sum_{a_1 \in T(r) \& a_2 \notin T(r)} [connected(a_1, a_2) + connected(a_2, a_1)] / |Arc| \quad . \tag{6}$$

Where connected(a_1, a_2) represents the coupling weight between a_1 and a_2, Arc stands for the set of arcs in the process model, and $|Arc|$ is the number of elements in Arc. The larger δ_r is, the higher the role activity coupling of r is.

2)The role coupling is not only related with role activity coupling, but also related with the number of roles associated with a role. If a role is associated with more roles, it may be complicated. So, the role relation coupling of r is defined as

$$\varepsilon_r = N_r / |R| \tag{7}$$

Where N_r is the number of roles associated with r, and $|R|$ represents the number of roles in the process. The larger ε_r is, the higher the role relation coupling of r is. The role coupling metric is defined as

$$Cou_r = \delta_r \times \varepsilon_r \tag{8}$$

The lower the role cohesion is and the higher the role coupling is, the more complex the role is. Therefore, the role complexity is defined as

$$C_r = Cou_r / Coh_r \tag{9}$$

As each role is different in importance, the role complexity of each role should be accompanied by the appropriate weight depending on its importance. Then according to the weight of each role and its role complexity, we can get the role complexity of a business process. The weight of each role can be defined as

$$W_r = \frac{1}{3} \times \left(\frac{t}{T} + \frac{time}{TIME} + \frac{\cos t}{COST} \right) \tag{10}$$

Where t, $time$ and $cost$ represent the number of activities, time and cost to perform activities by r separately. T, $TIME$ and $COST$ represent the number of activities, time and cost to perform activities of the business process separately. In the equation (10), 1/3 is to ensure the sum of weights of all the roles is 1. The role complexity of the business process is defined as

$$C = \sum_{r \in R} W_r \times C_r \tag{11}$$

Where C_r is the role complexity of the role r, W_r represents its weight, and R is the set of roles in the process.

3. The fitness function of role-oriented process mining

In 2005, Aalst first introduced genetic algorithm to process mining (genetic mining). In genetic mining, an individual is a candidate process model and the fitness function evaluates how well it is able to represent the actual process [6]. The fitness function is used to evaluate the adaptation of every individual, and guide searching process of genetic programming. In order to mine the simplified business process model, we introduce the complex fitness into the fitness function.

3.1 Role complexity fitness

We define $C(R)$ as role complexity of process individual, $min(C)$ and $max(C)$ stand for the minimum role complexity value and the maximum role complexity value separately

in a generation of population. So, the role complexity fitness is defined as

$$PF_{Complex}(R) = \frac{C(R) - \min(C)}{\max(C) - \min(C)} \quad (12)$$

$PF_{complex}$ describes the relative role complexity of individuals in the same population in (12). When the role complexity value of an individual is the maximum, the fitness value of role complexity is 1. When the role complexity value of individual reaches the minimum, the fitness value of role complexity is 0. The smaller $PF_{complex}$ of the individual is, the lower its relative complexity is.

3.2 Fitness function

The basic principle of fitness function is that a process model should match event logs as much as possible. So the precision is defined as:

$$PF_{precise}(R) = \sum_{i=1}^{|R|} \sum_{p_1 != p_2 \,\&\, p_1, p_2 \in f(R_i)} \cos(p_1, p_2) / 2 \quad (13)$$

Where $|R|$ means the number of roles in a process model, $f(R_i)$ is the participant set of R_i, and $cos(p_1,p_2)$ is the cosine similarity between the participant p_1 and p_2.

In order to discover simple process models, we add the role complexity fitness to describe the precision. As mining process models with correct roles is the nature of process mining, complex fitness should have lower weights. The individuals are punished that are complex. Assuming that the weight of the precision and complexity fitness are λ and θ separately, the complete fitness is defined as follows:

$$PF(R) = \lambda PF_{precise}(R) - \theta PF_{complex}(R) \quad (14)$$

where the fitness is affected by not only the correct recognition of roles in the process model, but also by the role complexity of the model. So, it can make the role complexity of mined process models lower.

The basic idea of genetic mining is as follows. First, event logs are collected and activities by each participant are analyzed. Then, initial population is created. After that, the fitness of every individual in the population is computed according to the fitness function （14）. If the fitness does not satisfy the termination condition, the population needs iterative evolution through the genetic operations including selection, crossover, and mutation. Each genetic operation transfers the individual, which has higher fitness value in the population to the next generation. This loop terminates until the optimal solution is found. In section 4, we resort to a case study to discuss the procedure of genetic mining in detail.

4. Case study

We first give a process mining experiment mentioned in [11] and compare its process mining algorithm with ours.

The interaction between roles and role identification are analyzed by using genetic algorithm and achieving optimal role identification [11]. On one hand, it shows the degree of similarity in the activities executed by participants. On the other hand, it indicates the similarity of performing internal activities of the participants of a certain role, and the interaction between participants. Table 1 shows the fragment of workflow logs. The data in the first column represents the process instance number, the second column represents activities, and the third column is the participant corresponding to the activity in the second column.

TABLE 1 The fragment of workflow log

Instance number	Activities	Participants
115	Business negotiations(a)	p101
115	Accept order(b)	p101
116	Business negotiations(a)	p107
115	Sample design(c)	p114

115	Sample inspection(d)	p114
116	Accept order(b)	p107
117	Business negotiations(a)	p101
115	Receiving materials(e)	p105
116	Receiving materials(e)	p105
115	Material application (f)	p105
115	Material confirmation(g)	p105
115	Parts production(h)	p106
115	Parts delivery(i)	p101
117	Business negotiations(a)	p101

The matrix *M* shows the role situation of a process model *Q*. If M_{jk} is 1, that means the participants *j* and *k* undertake the same role. If M_{jk} is 1 and *j* = *k*, that means *j* undertakes the role by himself[11]. As seen in matrix *M*, *p101, p105* and *p115* undertake a role, *p*106 undertakes a role alone, and *p*107 and *p*114 undertake a role. We can encode the process model *Q* through linking value of each row in *M*, and the chromosome is 01000100001100001000 0.

$$M = \begin{array}{c} \\ p101 \\ p105 \\ p106 \\ p107 \\ p114 \\ p115 \end{array} \begin{array}{c} \begin{array}{cccccc} p101 & p105 & p106 & p107 & p114 & p115 \end{array} \\ \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 1 \\ & 0 & 0 & 0 & 0 & 1 \\ & & 1 & 0 & 0 & 0 \\ & & & 0 & 1 & 0 \\ & & & & 0 & 0 \\ & & & & & 0 \end{pmatrix} \end{array}$$

In workflow logs, the situation of activities by each actor is as follows: *p101* executes *a* 10 times, *b* 6 times, and *i* 5 times separately. *p105* executes *c* 2 times, *d* 2 times, *e* 5 times, and *f* 6 times separately. *p106 e*xecutes *h* 8 times. *p107* executes activity a 8 times, *b* 7 times, and *i* 6 times separately. *p114* executes activity *c* 7 times, *d* 7 times, *e* 8 times, *f* 6 times, *g* 10 times, and *h* 2 times separately. *p115* executes *b* 5 times, *c* 3 times, *d* 3 times, *e* 5 times, *f* 4 times and *h* 1 times separately. Table 2 and table 3 show the data and abilities required for each activity in the process separately.

TABLE 2 Data required for each activity

activity	Data elements
sample design	Order contracts, sample instructions, sample
sample inspection	sample instructions, sample, sample inspection form
receiving	Order contracts, materials receipt form
materials	material application form, materials receipt form,
material	materials confirmation form, materials
application	material application form, materials receipt form,
material	materials confirmation form, materials
confirmation	materials, sample, materials confirmation form,
parts production	sample inspection form, finished parts
parts delivery	finished parts

TABLE 3 Abilities required for each activity

activity	Required ability
sample design	sample design ability, sample identification ability
sample inspection	sample identification ability
receiving materials	material identification ability
material application	material identification ability
material confirmation	material identification ability
parts production	production ability
parts delivery	delivery ability, negotiation ability

The precision value and the role complexity of *Q* represented in the matrix *M* are as follows:

$$PF_{Precise}(R) = \sum_{i=1}^{|R|} \sum_{p_1 != p_2 \& p_1, p_2 \in f(R_i)} \cos(p_1, p_2) / 2 | f(R_i) | \approx 1.33$$

$$C_R = C_{R1} + C_{R2} + C_{R3} \approx 392.19$$

The maximum role complexity value in this generation is 393.74, and the minimum one is 25.54. So the complex fitness value of *Q* is:

$$PF_{Complex}(R) = \frac{C(R) - \min(C)}{\max(C) - \min(C)} \approx 0.996$$

It is supposed that the complex fitness has lower weight than the precise fitness. In this paper, we assume that the weight of the precision fitness is 0.7 and that of the complex fitness is 0.3. So the fitness value of *Q* is

$PF(R) = \lambda PF_{Precise}(R) - \theta PF_{Complex}(R) = 0.632$ After that, genetic operations are performed: we use the

selection operator. It retains process models which have higher fitness values. Herein, we choose 15 process models to get corresponding chromosomes in each generation .Then, we use the crossover operator. For example, the model *A* with the code 010000000001000010001 makes a change in the 16th bit with the model *B* with the code 01000100001100011000, the new chromosomes 010000000001000110001 and 01000100001100001000 are produced. Its occurrence probability is $P_{c.}$

$$P_c = \frac{C_1(f_{\max} - f_m)}{f_{\max} - f_{\min}} + C_2 \approx 0.58$$

where C_1 and C_2 are constants, and we assume that they are 0.7 and 0.1. f_{max} and f_{min} are the maximum and minimum fitness value in this generation separately: 0.87 and 0.52. *fm* is the fitness value of *Q*, which is 0.632.

The mutation operation is that one bit of the chromosome changes at random, from 0 to 1 or from 1 to 0. Its probability of occurrence is 1 - (1 - P_m)5 and P_m is defined as

$$P_m = \frac{C_3(f_{\max} - f_m)}{f_{\max} - f_{\min}} + C_4 \approx 0.07$$

where C_3 and C_4 are constants and we assume that they are 0.09 and 0.01 separately.

As mentioned above, we can get the role situation matrix *MA* through role-oriented genetic mining below.

$$MA = \begin{array}{c} \\ p101 \\ p105 \\ p106 \\ p107 \\ p114 \\ p115 \end{array} \begin{array}{c} \begin{array}{cccccc} p101 & p105 & p106 & p107 & p114 & p115 \end{array} \\ \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 \\ & 0 & 0 & 0 & 1 & 1 \\ & & 1 & 0 & 0 & 0 \\ & & & 0 & 0 & 0 \\ & & & & 1 & 1 \\ & & & & & 0 \end{pmatrix} \end{array}$$

As seen from *MA*, *p101* and *p107* share a role, *p105*, *p114* and *p115* undertake a role, both *p106* and *p114* undertake a role alone. It groups the participants into four roles: Business manager, Technical staff(R_2), Technical staff(R_3) and Production workers. Figure 1 is the role-activity diagram through our role-oriented process mining method.

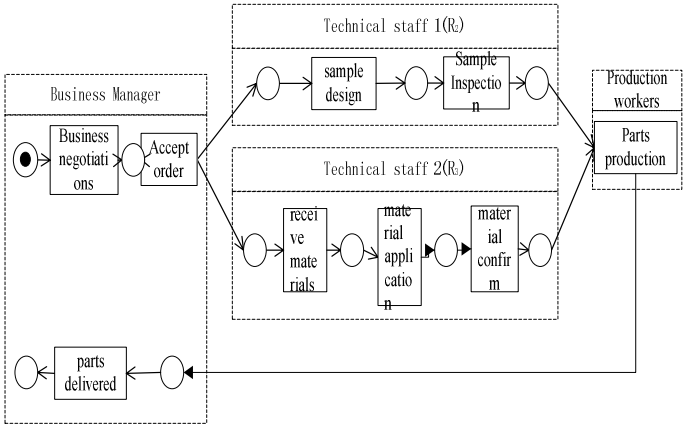

FIGURE 1: The mined role-activity diagram

In order to verify the effectiveness of our method, we compare the role complexity of process models mined by our algorithm and the algorithm proposed in [11]. For Figure 1, we can calculate the role complexity of R_2 and R_3 by our algorithm. The cohesion and coupling complexity of R_2 and R_3 are shown in table 4.

TABLE 4 Cohesion and coupling complexity

	R_2	R_3
role activity cohesion	1	5/6
role data cohesion	3/4	1/2
role ability cohesion	1	4/3
role activity coupling	2/9	2/9
role relation coupling	1/2	1/2
role complexity	0.15	0.20

R_2 and R_3 are taken as one role [11]. The role complexity of that role is 7.82. Clearly, the role complexity of that role is far greater than the sum of the role complexity of R_2 and R_3 in Figure 1. The reason is that the technical staff is mainly responsible for designing samples and inspection, as well as raw material application and confirmation. These two kinds of activities are different and require different data and abilities, which leads to low role activity cohesion, role data cohesion and

role ability cohesion. This brings about the high role complexity ultimately. By means of our method, the work of technical staff is split, which ensures the process model is correct and has low role complexity at the same time. The result shows that our method performs better in discovering simplified role-based process models. In fact, the roles have high cohesion and low coupling.

The role mining algorithms proposed in [12] and [13] got role hierarchy through the combination of permissions based on participants and their permissions. Their algorithm identified roles based on permissions, ignorant of the difference between the activities of participants. Keith and Martin measured the role's complexity through surveying internal activities and interactions between roles [14]. They didn't give full considerations of cohesion and coupling between roles. In addition, [15] considered the complexity of the application of resources. But it ignored the internal cohesion of roles. In comparison, our method treats the similarity between activities by different participants as the basis for identifying roles and it is based on genetic mining. So it deals with noise more effectively in workflow logs. Additional, it measures the complexity of roles through cohesion and coupling in terms of activities and resources. Therefore, the role complexity makes the process model correct and simple.

5. Experiments

In order to analyze the performance of the algorithm we proposed, we select some event logs produced by 8 workflow models shown partly in Table 1 to perform some experiments.

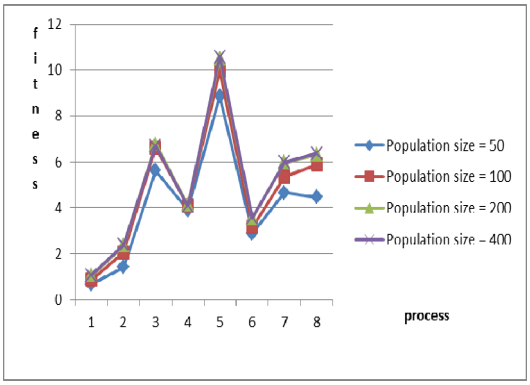

FIGURE 2. Comparison of different population size

As can be seen from Figure 2, when the population size is small, the fitness value is low. And when the population size is bigger than 200, the fitness value no longer increases. So, the population size is set 200.

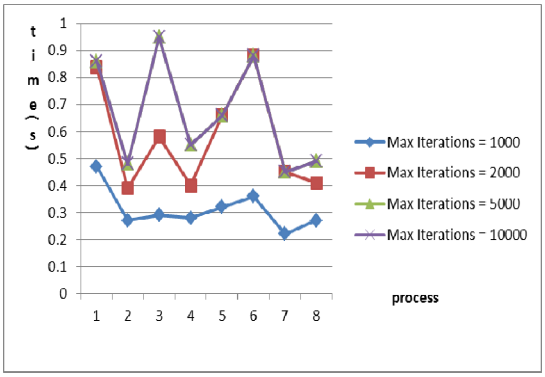

FIGURE 3: Comparison of different maximum iterations

As shown in Figure 3, the time spent by the algorithm is increased with the increase of the maximum number of iterations. When the maximum number of iterations is small, the algorithm will stop before finding the optimal solution. In this case, the solution is questionable. And when the maximum number of iterations reaches 5000, the time spent by the algorithm will remain stable. That means the optimal solution will be found before 5000 iterations. So, the maximum number of iterations is set to 5000.

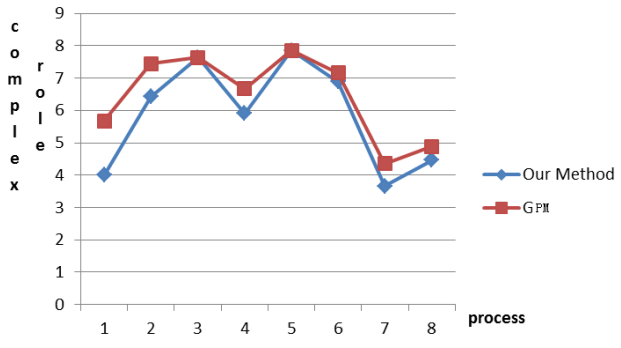

FIGURE 4. Comparison of role complexity

Except process 3 and 5, the role complexity of process models mined by our method is lower than that by genetic process mining (GPM) [4] in Figure 4. The reason is that the role complexity of process models is not considered in GPM [4]. And in process 3 and 5, the role complexity of process models may be not reduced any more. Therefore, the algorithm we propose can reduce the role complexity of mined process models.

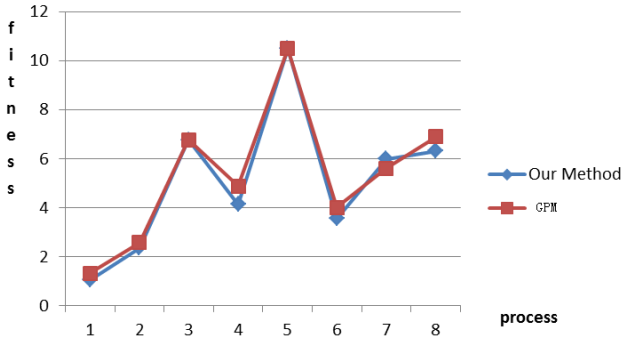

FIGURE 5. Comparison of fitness

In Figure 5, we can see that the fitness value of process models mined by our method is relatively close. That means, though our method considers role complexity, it does little adverse effect on the fitness.

Through these experiments, we can see that the algorithm performs better when mining simpler process models, because it uses the role complexity of process models. Therefore, it can reduce the role complexity when mining process models.

6. Conclusions

In this paper, we combine genetic programming with the role complexity, and propose the role-oriented process mining approach. The advantage of our method is that it can mine process models not only correctly, but also simply. In the future, we will consider the relationship between process roles more comprehensively and reduce the role complexity further. Besides, we can improve the efficiency of model mining through improved genetic algorithm.

Acknowledgments

The National Nature and Science Foundation of China under Grants nos. 71071038 supports this work. Many thanks to Hongzhi Hu for her assistant work to the corresponding author Weihui Dai of this article.

References

[1] J E. Cook. Process Discovering and validation through event-data analysis. Boulder, University of Colorado, *Technical Report*: CU-CS-817-96, 1996

[2] R Agrawal, D Gunopulos, and F. Leymann, ”Mining process models from workflow logs,” in *Proceedings of the 6th International Conference on Extending Database Technology*, pp.469-483, 1998.

[3] W.M.P. van der Aalst，H.A. Reijers，and A.J.M.M. Weijters, “Business process mining: an industrial application,” *Information Systems* ， vol.32, no.5, pp. 713-732, 2007

[4] A. de Medeiros, A.K. Weijters, and A.J.M.M., et al, “Genetic process mining: an experimental Evaluation,” *Journal of Data Mining and Knowledge Discovery, vol.14,no.2,pp.* 245-304, 2007.

[5] K. Tian，and Z. Qingxin, “Study of workflow reconstruction based on hybrid genetic algorithm,” *Computer Science,* vol.34, no.1,pp. 103-105, 2007.

[6] K.B.Lassen, and W.M.P. van der Aalst, “Complexity metrics for workflow nets,” *Information and Software Technology*, vol.51,no.3, pp. 610-626, 2009

[7] C. A. Ellis, A. J. Rembert, and, Kwang-Hoon Kim, et al, “Beyond workflow mining,” in*Proceedings of BPM 2006*, pp. 49-64, 2006

[8] M. Song, and W.M.P. van der Aalst, “Towards comprehensive support for organizational mining,” *Decision Support Systems,vol.46,no.1,pp.*300-317,2008

[9] J. Cardoso, “Process control-flow complexity metric: an empirical validation,” in *Proceedings of the IEEE International Conference on Services Computing*, pp. 167-173, 2006

[10] I. Vanderfeesten, H. A. Reijers, and Wil M. P. van der Aalst, “Evaluating workflow process designs using cohesion and coupling Metrics,” *Computers in Industry*, vol.59, no.5, pp. 420-437, 2008

[11] W. D. Zhao. W. H. Dai, and A. H. Wang, et al, “Role-activity diagrams modeling based on workflow mining,” in *Proceedings of Computer Science and Information Engineering*, pp. 301-305, 2009

[12] M. Kuhlmann, D. Shohat, and G. Schimpf, “Role mining-revealing business roles for security administration using data mining technology,” in *Proceedings of the 8th ACM Symposium on Access Control Models and Technologies*, pp. 179-186, 2003

[13] S.Jürgen, and S.Ulrike, “Role mining with ORCA,” in *Proceedings of the tenth ACM Symposium on Access control models and technologies*, pp. 168-176, 2005

[14] H. A. Reijers, and I Vanderfeesten, “Cohesion and coupling metrics for workflow process design,” in Proceedings of Second International Conference on Business Process Management, pp. 290-305, 2004.

[15] A. Kumar, R.M. Dijkman, and M. Song, “Optimal resource assignment in workflows for maximizing cooperation,” in Proceedings of 11th International Conference on Business process management, pp.235-250, 2013.