**TELKOMNIKA Telecommunication, Computing, Electronics and Control**
Vol. 18, No. 1, February 2020, pp. 106~113
ISSN: 1693-6930, accredited First Grade by Kemenristekdikti, Decree No: 21/E/KPT/2018
DOI: 10.12928/TELKOMNIKA.v18i1.12982 

# Graph-based algorithm for checking wrong indirect relationships in non-free choice

**Agung Wiratmo, Kelly Rossa Sungkono, Riyanarto Sarno**
Department of Informatics, Faculty of Information Technology and Communication,
Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia

**Article Info**

***Article history:***

Received May 15, 2019
Revised Jun 14, 2019
Accepted Jul 1, 2019

***Keywords:***

Decision mining
Parameterize decision mining
Process model
Wrong indirect relationships

**ABSTRACT**

In this context, this paper proposes a combination of parameterised decision mining and relation sequences to detect wrong indirect relationship in the non-free choice. The existing decision mining without parameter can only detect the direction, but not the correctness. This paper aims to identify the direction and correctness with decision mining with parameter. This paper discovers a graph process model based on the event log. Then, it analyses the graph process model for obtaining decision points. Each decision point is processed by using parameterised decision mining, so that decision rules are formed. The derived decision rules are used as parameters of checking wrong indirect relationship in the non-free choice. The evaluation shows that the checking wrong indirect relationships in non-free choice with parameterised decision mining have 100% accuracy, whereas the existing decision mining has 90.7% accuracy.



***Corresponding Author:***

Riyanarto Sarno,
Department of Informatics, Faculty of Information Technology and Communication,
Institut Teknologi Sepuluh Nopember,
Surabaya, Indonesia.
Email: riyanarto@if.its.ac.id

## 1. INTRODUCTION

Each company records events carried out in the event log. The analysation of information from the event log obtains the obtain knowledge [1]. The process of gaining knowledge from event log extraction is called the process mining, which aims to find out, monitoring and improving the processes that occur [2]. In the process mining, there are two most prominent processes, namely: 1) conformance checking [3] and 2) process discovery [4]. The process mining founds the wrong process in the event log. This paper discovers a graph process model based on the event log. The analysation of the graph process model obtains decision points. Each decision point is processed by using parameterised decision mining, so that decision rules are formed. The derived decision rules are used as parameters of checking wrong indirect relationship in the non-free choice.

Several previous studies discuss decision mining in recent years. A study conducted by Rozinat [5] explains decision mining on business processes. However, decision mining is not implemented as a decision rule for checking errors in event logs. Horita [6] made decisions on event logs that result in linear temporal logic, but the temporal logic is not applied for error searches in event logs. The existing methods in checking indirect relationships [7, 8] in non-free choice only using direction so the error can be detected only from directional error, but correctness from choosing directional cannot be obtained. The proposed method, namely parameterized decision mining is to use the decision rule in checking event logs with notice not only from

the direction but also the parameters in the event log. In this research, the decision rule is used to find errors in the event log, so the failure of event logs can be finding more accurate in terms of the direction and correctness of choice.

## 2. RESEARCH METHOD

In this section, parameterized decision mining will be presented to find a wrong indirect relationship in non-free choice using the graph database. The non-free choice is a condition that is not free to make choices, but choices depend on the results of the previous election [9]. Checking process models on the non-free choice part of the event log can be done correctly must consider all dependencies [10]. There are two types of dependence on the process of the model, direct dependence or referred to as direct relationship and indirect dependence or referred to an indirect relationship [7, 11, 12]. A direct relationship is a relationship or dependency that is directly between tasks. Conversely, an indirect relationship is a relationship or a dependence that is indirectly between tasks [13]. A graph database is a NoSQL database where is depicted in the form of graph [14-16]. Graph databases will form data as nodes and relations between nodes [15, 17]. The process mining is used to extract information from the event log to see business processes [10, 18, 19]. The process mining can be used to build a process model [16, 20–23]. Decision mining is used to study parameters that can influence the selection of grooves [24, 25].

Decision mining is used to find rules for branching from each decision point. By using a graph database, a decision point can be known from a node that has a xorsplit relation. The algorithm used in decision mining research is using the C4.5 decision tree algorithm [5, 26]. A decision tree is used to predict an activity seen from the parameters of a data. The decision tree has several terms. Those terms are a root as the initial node, a leaf node as the child of a node, and the depth of a node as the length of the path between the nodes to the leaf node [27, 28]. The first step is to discover graph process model of the event log based on the graph database. Then, graph process model be analyzed to find the decision point. The second step is discovering decision rule using decision mining from each decision point with notice parameter in event log. This decision rule will be used as a parameter in determining the wrong decision in non-free choice. The last step searches each case in the event log with the parameters stated earlier.

### 2.1. Discovery process model based on graph database

The first step for the discovery of the graph model process is to enter event logs like in Table 1 into the graph database using a query like in Table 2. The parameters in the event log used in the graph database are Case_ID, Activity, and Time. In the Table 2 shown queries in the graph database for: (1) Import all data in event log, (2) Import only unique activity, (3) create relation: sequence, xorsplit, xorjoin, andsplit, andjoin, non-free choice, and (4) get the decision point.

In the model process, there will be several relationships such as xorsplit [29], xorjoin [29], andsplit [30], andjoin [30] and non-free choice shown in Figure 1. Joint relations are a relation of the union from branching to split relations at the base of branching. The xor relation is a branching relation which means that the flow of the event log can only choose one of the entire branches of the event log. The and relation is a flow relationship that will do all events even though the different order. The results of working on queries in Table 2 produce the model process shown in Figure 1. After each activity is represented in the node, the next step is to make relations between nodes such as sequence, xorsplit, xorjoin, andjoin, andsplit relations, and non-free choice.

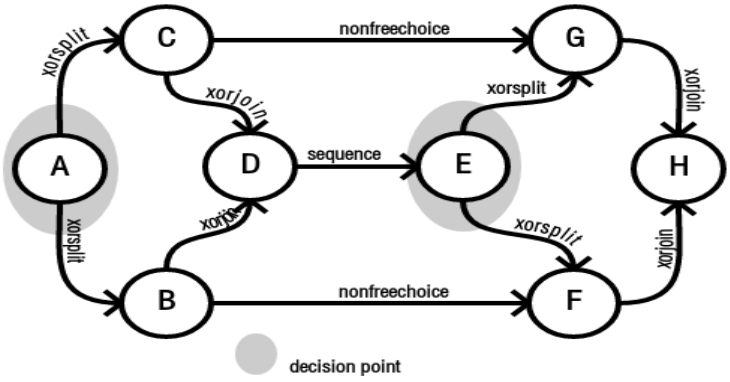


Figure 1. Examples of the process model

After the graph process model is formed. The next step is to determine the decision point by using the query in Table 2. Decision point is the node where branching of the process begins. In Figure 1 shows graph process model where containing the decision points in node A and node E. Node A is the base of member node B or node C. Node E is the base of branching node G and node F. furthermore, decision rule would be discovered by decision mining to find parameters each branching of each decision point.

### 2.2. Extracting a parameter in decision points

The decision point shown in Figure 1, there are two decision point points. Each decision point will be analyzed by considering the parameters in the event log to get the decision rule. The process for extraction decision rule is called decision mining. The algorithm of decision mining is used C4.5 decision tree algorithm. The first step is getting leaf nodes from each decision point. Then, the next step is getting an event log that has activities such as leaf nodes with algorithm in Table 3.

The data needed in algorithm in Table 3 is $decisionPoint, leafNode,$ dan $eventLog$. Each decision point of leaf nodes obtained from algorithm in Table 3 is used in the decision mining process. The decision mining algorithm is seen in Table 4. Algorithm in Table 4 has five data variables: X is the event log on node leaf. Y is an attribute owned by X. The $splitingAtribute$ is an attribute used as a solving parameter. The $atributeSelectionMethod$ is the method used to find the best fraction value. The method used to find the best splitting criterion is C4.5 with the gini index parameter in (2) and (3). The function of (1) is to evaluate the separation in event log each attribute. The function of (2) is to evaluate the separation in event log each parameter.

$$Gini(t) = 1 - \sum[p(\frac{j}{t})]^2 \quad (2)$$

where $p(\frac{j}{t})$ represents the frequency of the j attribute in activity t.

$$Gini_{split} = \sum_{i=1}^{k} \frac{n_i}{n} Gini(i) \quad (3)$$

where $k$ is the number of partitions, $n_i$ is the amount of data in $i$ partition, $n$ is the amount of data in the $p$ node. The best split value is indicated by the smallest $Gini_{split}$. The $splitingCriteria$ is the value of the parameter that is used as a solver. $N$ is a node. The algorithm in Table 4 will continue to be repeated until the data is $X$ empty. The results of algorithm in Table 4 are a decision tree by showing the parameters and the leaf values can be seen in Figure 2.

Table 1. Process mining will be carried out by the event log

| Case_ID | amount | stackType | Status | Time | Activity |
|---|---|---|---|---|---|
| PP10 | 3 | nonreefer | complete | 3/11/2016 0:52 | A |
| PP10 | 3 | nonreefer | complete | 3/11/2016 2:00 | C |
| PP10 | 3 | nonreefer | complete | 3/11/2016 3:08 | D |
| PP10 | 3 | nonreefer | complete | 3/11/2016 4:16 | E |
| PP10 | 3 | nonreefer | complete | 3/11/2016 5:24 | F |
| PP10 | 3 | nonreefer | complete | 3/11/2016 6:32 | H |
| PP412 | 17 | reefer | incomplete | 7/2/2016 22:28 | A |
| PP412 | 17 | reefer | incomplete | 7/2/2016 23:36 | C |
| PP412 | 17 | reefer | incomplete | 7/3/2016 0:44 | D |
| PP412 | 17 | reefer | incomplete | 7/3/2016 1:52 | E |
| PP412 | 17 | reefer | incomplete | 7/3/2016 3:00 | G |
| PP412 | 17 | reefer | incomplete | 7/3/2016 4:08 | H |
| PP735 | 13 | nonreefer | incomplete | 10/2/2016 10:52 | A |
| PP735 | 13 | nonreefer | incomplete | 10/2/2016 12:00 | B |
| PP735 | 13 | nonreefer | incomplete | 10/2/2016 13:08 | D |
| PP735 | 13 | nonreefer | incomplete | 10/2/2016 14:16 | E |
| PP735 | 13 | nonreefer | incomplete | 10/2/2016 15:24 | G |
| PP735 | 13 | nonreefer | incomplete | 10/2/2016 16:32 | H |
| PP1050 | 10 | nonreefer | complete | 12/30/2016 16:52 | A |
| PP1050 | 10 | nonreefer | complete | 12/30/2016 18:00 | B |
| PP1050 | 10 | nonreefer | complete | 12/30/2016 19:08 | D |
| PP1050 | 10 | nonreefer | complete | 12/30/2016 20:16 | E |
| PP1050 | 10 | nonreefer | complete | 12/30/2016 21:24 | F |
| PP1050 | 10 | nonreefer | complete | 12/30/2016 22:32 | H |

Table 2. Queries in the graph database

| No | Queries |
|---|---|
| 1 | def importActivityT (tx, fileName):<br>tx.run ("LOAD CSV with headers FROM 'file:///"+fileName+"' AS line "<br>"Merge (: Activity {CaseId: line. Case_ID, Name: line. Activity, Amount: toInt (line. amount), StackType: line. stackType, Status: line. status, Time: line. Time})") |
| 2 | def importCaseActivity (tx, fileName):<br>tx.run ("LOAD CSV with headers FROM 'file:///"+fileName+"' AS line Merge (: CaseActivity {Name: line. Activity}) “) |
| 3 | def createRelationship(tx):<br># create sequence relation<br>tx.run ("MATCH (c: Activity) "<br>"WITH COLLECT(c) AS Caselist "<br>"UNWIND RANGE (0, Size (Caselist) - 2) as idx "<br>"WITH Caselist[idx] AS s1, Caselist[idx+1] AS s2 "<br>"MATCH (b: CaseActivity), (a: CaseActivity) "<br>"WHERE s1. CaseId = s2. CaseId AND s1. Name = a. Name AND s2. Name = b. Name "<br>"MERGE (a)- [r: SEQUENCE]->(b)")<br># create xorsplit relation<br>tx.run ("MATCH (bef)-[r]->(aft) "<br>"WHERE size((bef)--> ())>1 AND size((aft)<--()) =1 AND (size((aft)--> ()) =1 OR size((aft)--> ())>1) "<br>"CREATE (bef)- [: XORSPLIT]->(aft) "<br>"DELETE r")<br># create xorjoin relation<br>tx.run ("MATCH (bef)-[r]->(aft) "<br>"WHERE (size((bef)--> ()) =1 OR size((bef)--> ())>1) AND size((aft)<--())>1 "<br>"CREATE (bef)- [: XORJOIN]->(aft) "<br>"DELETE r")<br># create andsplit relation<br>tx.run ("MATCH (aft1) <-[r]-(bef)-[s]->(aft2)"<br>"WHERE size((bef)--> ())>1 "<br>"AND size((aft2) --> ()) =size((bef)--> ()) AND size((aft1) --> ()) =size((bef)--> ()) "<br>"AND not (aft1)- [: SEQUENCE]->(bef) AND not (aft2)- [: SEQUENCE]->(bef) "<br>"MERGE (aft1) <- [: ANDSPLIT] -(bef)- [: ANDSPLIT]->(aft2) "<br>"DELETE r, s")<br># create andjoin relation<br>tx.run ("MATCH (aft1)-[r]->(bef)<-[s]-(aft2) "<br>"WHERE size((bef)<--())>1 "<br>"AND size((aft2) --> ()) =size((bef)<--()) AND size((aft1) --> ()) =size((bef)<--()) "<br>"AND not ()- [: ANDSPLIT]->(bef) "<br>"MERGE (aft1)- [: ANDJOIN]->(bef)<-[: ANDJOIN]-(aft2) "<br>"DELETE r, s")<br># create Non-Free Choice<br>tx.run ("match ()- [c: XORSPLIT]->(n) "<br>"match (a)- [b: XORJOIN]-> () "<br>"match (k: Activity), (l: Activity) "<br>"where a. Name<>n.Name and k. Name=a.Name and l. Name=n.Name and k. CaseId=l.CaseId and k. Time<l.Time "<br>"merge (a)- [: NONFREECHOICE]->(n)") |
| 4 | def printStartingNodeNonFreeChoice(tx):<br>nodes = []<br>global nodeStartedNonFreeChoice<br>for record in tx.run ("MATCH (p)- [r: XORSPLIT]-> () RETURN p. Name ORDER BY p. Name"): nodes. append (record ["p. Name"])<br>nodeStartedNonFreeChoice = np. unique (np. array(nodes))<br>return nodeStartedNonFreeChoice |

Table 3. Algorithm for getting event log of leaf nodes each decision point

Data: $decisionPoint, leafNode, eventLog$
Result: $Event\ log\ pada\ leaf\ node$

| No | Pseudocode |
|---|---|
| 1 | $DataLeaf$ |
| 2 | $for\ each\ x\ of\ decisionPoint\ do$ |
| 3 | $for\ each\ y\ of\ leafNode\ of\ each\ x\ do$ |
| 4 | $for\ z\ of\ eventLog\ do$ |
| 5 | $if\ activity\ in\ each\ z\ of\ trueEventLog\ equals\ with\ activity\ in\ y\ leafNode\ then$ |
| 6 | $attach\ a\ event\ Log\ z\ to\ DataLeaf$ |
| 7 | $end$ |
| 8 | $end$ |
| 9 | $end$ |
| 10 | $end$ |

Table 4. Algorithm for extracting a parameter each decision point using decision mining

Data: X, Y, splitingAtribute, atributeSelectionMethod, splitingCriteria, N
Result: Parameter decision in decision point showed by decision tree

| No | Pseudocode |
|---|---|
| 1 | *Create Node N* |
| 2 | *if tuple in X are all of the same activity then* |
| 3 | *return N as a leaf node labeled with activity in X* |
| 4 | *end* |
| 5 | *if Y is empty then* |
| 6 | *return N as a leaf node labeled with majority activity in X* |
| 7 | *end* |
| 8 | $atributeSelectionMethod(X, Y)$ |
| 9 | $if\ (splitingAtribute\ is\ discrete - valued)\ then$ |
| 10 | $Y \leftarrow Y - splitingAtribute$ |
| 11 | *end* |
| 12 | *for each outcome i of splitingCriteria do* |
| 13 | *let Xi be the set of data tuples in X satisfying the outcome j* |
| 14 | *if Xi is empty then* |
| 15 | *attach a leaf labeled with majority activity in X to node N* |
| 16 | *else* |
| 17 | $atributeSelectionMethod(Xi, Y)$ |
| 18 | *end* |
| 19 | *end* |
| 20 | *return N* |

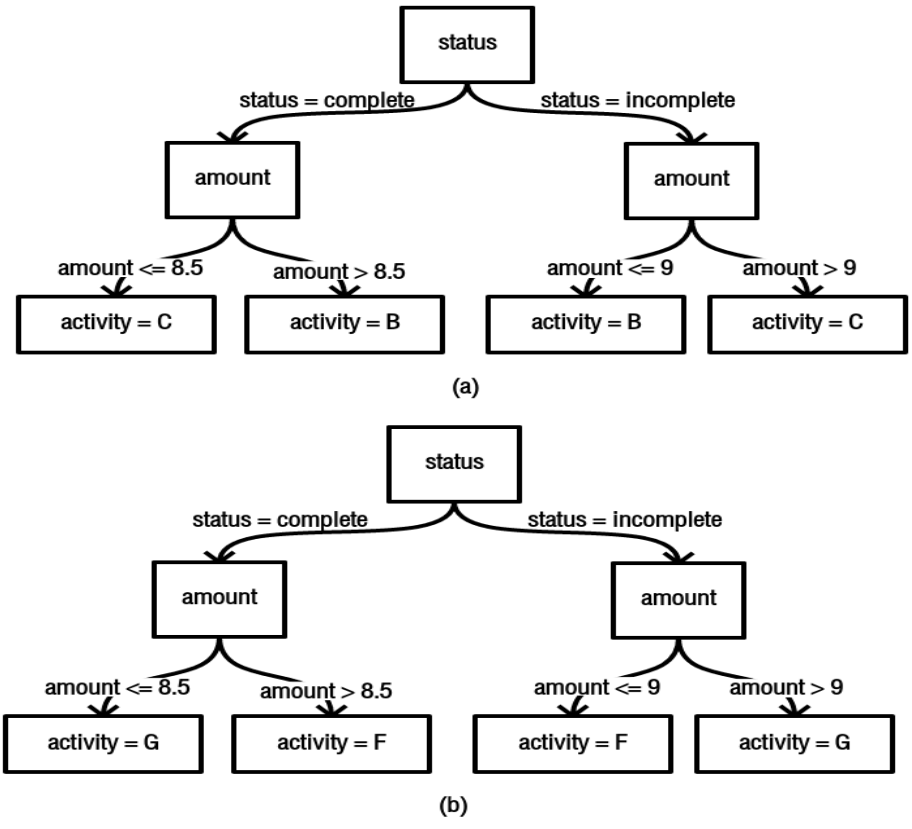


Figure 2. Decision tree in: (a) node A, and (b) node E

From Figure 2 (a) it can be concluded that the condition for doing activity B is ((status = complete AND amount > 8.5) OR (status = incomplete AND amount ≤ 9)). Whereas to do activity C it must be conditioned ((status = complete AND amount ≤ 8.5) OR (status = incomplete AND amount > 9)). From Figure 2 (b) it can be concluded that the condition for doing activity F is ((status = complete AND amount > 8.5) OR (status = incomplete AND amount ≤ 9)). Whereas to do activity G must be conditioned ((status = complete AND amount ≤ 8.5) OR (status = incomplete AND amount t> 9)). The next step is to check the event log with parameters that have been obtained previously.

### 2.3. Checking wrong indirect relationships

After finding these parameters, then looking for wrong indirect relationship in non-free choice. Checking is done in each case. This is because each case has different parameters. The goal is to find faults with precession far better than just using the direction of each case. Checking scheme for non-free choices containing indirect relations as in algorithm in Table 5. From the algorithm in Table 5, there are needed several parameters like decisionParameter and $EventLog$. The process will repeat as many cases as in EventLog and checking in case i with $decisionParameter$. If the process is not same with the rule, then the case i will be added in $IndirectRelationship$.

Table 5. Algorithm for checking indirect relationship in non-free choice

| Data: decisionParameter, EventLog<br>Result: IndirectRelationship | |
|---|---|
| No | Pseudocode |
| 1 | IndirectRelationship |
| 2 | for each case i of EventLog do |
| 3 | if case i parameter not equal with decisionParameter then |
| 4 | attach a case i to node IndirectRelationship |
| 5 | end |
| 6 | end |

## 3. RESULTS AND ANALYSIS

The proposed method is implemented in 1199 cases in event log. The event log has many various of attribute in parameter like amount, stackType, status, and times as in Table 1. From the 1199 case activity, the results of checking using the proposed method can be seen in Figure 3. From Figure 3 (a) having the sequence of events A → C → D → E → F → H is the sequence of events that are wrong based on the parameters and based on the order of non-free choice. Case_ID PP10 has several parameters, which value of parameter status is complete and in parameter amount has 3. In Figure 3 (a) shows after activity A goes to activity C is a correct but wrong decision after activity E goes to activity F.

Figure 3 (b) has the order of events A → C → D → E → G → H is the sequence of events incorrectly based on parameters due to incomplete and amount 7 status parameters. In decision parameters obtained from decision mining requires that it can pass activity C then conditions fulfilled ((status = complete AND amount 8.5) OR (status = incomplete AND amount > 9)) and conditions for passing activity G must meet the requirements ((status = complete AND amount 8.5) OR (status = incomplete AND amount > 9)). However, when viewed based on the order non-free choice is correct.

Figure 3 (c) has the sequence of events A → B → D → E → G → H is the sequence of events that are wrong based on the parameters and based on the order of non-free choice. Case_ID PP735 has several parameters, which value of parameter status is incomplete and in parameter amount has 13. In Figure 3 (a) shows after activity A goes to activity B is a correct but wrong decision after activity E goes to activity G.

Figure 3 (d) has the order of events A → B → D → E → F → H is the sequence of events wrong based on parameters because complete status parameters and amount 3. In the decision parameters obtained from decision mining requires that it can pass activity B then the conditions fulfilled ((status = complete AND amount > 8.5) OR (status = incomplete AND amount 9)) and the condition for passing activity F must meet the requirements ((status = complete AND amount > 8.5) OR (status = incomplete AND amount 9)) . However, when viewed based on the order non-free choice is correct.

Accuracy of the existing method shows 900 case activity in $TP$, 188 case activity in $TN$, 111 case activity in $FP$ dan zero activity in $FN$.

$$Accuracy = \frac{900+188}{900+188+111+0}\ x\ 100\%$$

$$Accuracy = 0.907\ x\ 100\%$$

$$Accuracy = 90.7\%$$

The accuracy of the parameterize decision mining shows 900 case activity in $TP$, 299 case activity in $TN$, zero case activity in $FP$ dan zero activity in $FN$.

$$Accuracy = \frac{900+299}{900+299+0+0}\ x\ 100\%$$

$$Accuracy = 1\ x\ 100\%$$

$$Accuracy = 100\%$$

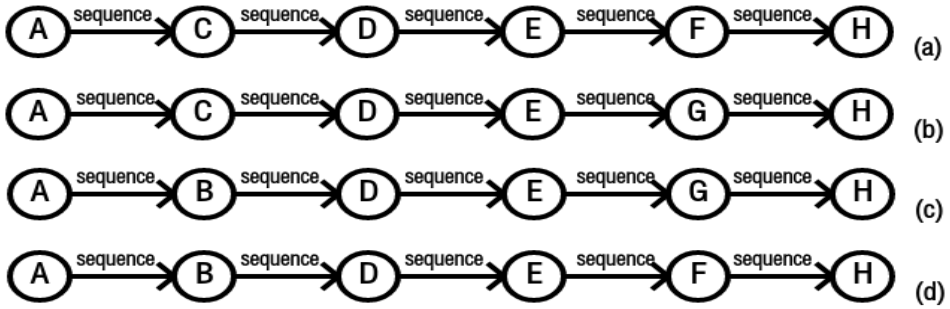


Figure 3. The result of the process which contains wrong indirect relationship in Case_ID: (a) PP10, (b) PP412, (c) PP735, and (d) PP1050

## 4. CONCLUSION

The decision mining overcomes the correct flow in a direction but not in a parameterize direction. The parameterized decision mining considers parameters in the selection of grooves. This paper proposes a combination of parameterized decision mining and relation sequences to detect the direction and correctness. Firstly, discovering a graph process model based on the event log. Then, an analysis of the graph process model obtains decision points. The process of each decision point is using parameterized decision mining, so that decision rules are formed. The derived decision rules are used as parameters of checking wrong indirect relationship in the non-free choice. The accuracy of the parameterized decision mining reaches 100%. It means the proposed method can detect errors far more precisely than the existing method only get 90.7% accuracy.

## REFERENCES

[1] Tax N., Sidorova N., Van Der Aalst W. M. P., "Discovering more precise process models from event logs by filtering out chaotic activities," *Journal of Intelligent Information Systems*, vol. 52, no. 1, pp. 107–139, 2019.
[2] Van Der Aalst W. M. P., "Decomposing Petri nets for process mining: A generic approach," *Distributed and Parallel Databases*, vol. 31, no. 4, pp. 471–507, 2013.
[3] Aleem S., Fernando Capretz L., Ahmed F., "Business Process Mining Approaches: A Relative Comparison," *International Journal of Science, Technology & Management*, vol. 4, no. 1, pp. 1557–1564, 2015.
[4] Van der Aalst W., "Process Discovery: An Introduction," *Process Mining-Springer*, pp. 125–156, 2011.
[5] Rozinat A, Van der Aalst W. M. P., "Decision Mining in ProM," *International Conference on Business Process Management*, pp. 420–425, 2006.
[6] Horita H., Hirayama H., Hayase T., Tahara Y, Ohsuga A., "Process mining approach based on partial structures of event logs and decision tree learning," *Proceedings - 2016 5th IIAI International Congress on Advanced Applied Informatics, IIAI-AAI*, 2016.
[7] Wen L., Wang J., Sun J., "Detecting implicit dependencies between tasks from event logs," *Asia-Pacific Web Conference*, pp. 591–603, 2006.
[8] Van Der Aalst W. M. P., De Medeiros A. K. A., "Process mining and security: Detecting anomalous process executions and checking process conformance," *Electronic Notes in Theoretical Computer Science*, vol. 121, pp. 3-21, 2005.
[9] Sungkono K. R., Sarno R., "Constructing control-flow patterns containing invisible task and non-free choice based on declarative model," *International Journal of Innovative Computing, Information and Control*, vol. 14, no. 4, pp. 1285-1299, 2018.
[10] Wen L., Van Der Aalst W. M. P., Wang J., Sun J., "Mining process models with non-free-choice constructs," *Data Mining and Knowledge Discovery*, vol. 15, no. 2, pp. 145–180, 2007.
[11] Kalynychenko O., Chalyi S., Bodyanskiy Y., Golian V., Golian N., "Implementation of search mechanism for implicit dependences in process mining," *Proceedings of the 2013 IEEE 7th International Conference on Intelligent Data Acquisition and Advanced Computing Systems*, 2013.
[12] Van Der Aalst W. M. P., De Beer H. T., Van Dongen B. F., "Process Mining and Verification of Properties: An Approach based on Temporal Logic," *Proceedings of the 2005 Confederated International Conference on On the Move to Meaningful Internet Systems*, pp. 130–147, 2005.

[13] Chabrol M., Dalmas B., Norre S., Rodier S., "A process tree-based algorithm for the detection of implicit dependencies," *2016 IEEE Tenth International Conference on Research Challenges in Information Science (RCIS)*, 2016.
[14] Joishi J., Sureka A., "Graph or Relational Databases: A Speed Comparison for Process Mining Algorithm," *ArXiv*, 2016.
[15] Kushwaha A., Pandey R.S., "A Graph Based Approach To Identify Objects Using Identifying Attribute," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 6, no. 2, pp. 438-446, 2017.
[16] Patil N. S., Kiran P., Kiran N. P., Naresh P. K. M., "A Survey on Graph Database Management Techniques for Huge Unstructured Data," *International Journal of Electrical and Computer Engineering*, vol. 8, no. 2, pp. 1140-1149, 2018.
[17] Sarno R., Sungkono K., Johanes R., Sunaryono D., "Graph-Based Algorithms for Discovering a Process Model Containing Invisible Tasks," *International Journal of Intelligent Engineering and Systems*, vol. 12, no. 2, pp. 85-94, 2019.
[18] Iwasaki K., Kuriyama Y., Kondoh S., Shirayori A., "Structuring engineers' implicit knowledge of forming process design by using a graph model," *Procedia CIRP*, vol. 67, pp. 563–568, 2018.
[19] Chapela-Campa D., Mucientes M., Lama M., "Mining Frequent Patterns in Process Models," *Information Sciences*, vol. 472, pp. 235–257, 2019.
[20] Caesarita Y., Sarno R., Sungkono K. R., "Identifying bottlenecks and fraud of business process using alpha ++ and heuristic miner algorithms (Case study: CV. Wicaksana Artha)," *2017 11th International Conference on Information & Communication Technology and System*, 2017.
[21] Bolt A., de Leoni M., van der Aalst W. M. P., "Process variant comparison: Using event logs to detect differences in behavior and business rules," *Information Systems*, vol. 74, pp. 53–66, 2018.
[22] Dakic D., Stefanovic D., "Business Process Mining Application: A Literature Review," *Proceedings of the 29th International DAAAM Symposium*, pp. 0866–0875, 2018.
[23] Rezaiee A. M., Karimi A., "A New Dynamic Intelligent Model to Determine Reliability and Trust of Online Banking by Using Fuzzy C-Mean," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 4, no. 3, pp. 605-610, 2016.
[24] Ait-Mlouk A., Agouti T., Gharnati F., "Mining and prioritization of association rules for big data: multi-criteria decision analysis approach," *Journal of Big Data*, vol. 4, no. 42, pp. 1-21, 2017.
[25] Mannhardt F., De Leoni M., Reijers H. A., Van Der Aalst W. M. P., "Decision Mining Revisited-Discovering Overlapping Rules," *International Conference on Advanced Information Systems Engineering*, pp. 377-392, 2016.
[26] Singh D., Choudhary N., Samota J., "Analysis of Data Mining Classification with Decision tree Technique," *Global Journals Inc. (USA)*, vol. 13, pp. 1-5, 2013.
[27] Bombara G., Vasile C-I, Penedo F., Yasuoka H., Belta C., "A Decision Tree Approach to Data Classification using Signal Temporal Logic," *Proceedings of the 19th International Conference on Hybrid Systems: Computation and Control - HSCC '16*, 2016.
[28] Saettler A., Laber E, de A., Mello Pereira F., "Decision tree classification with bounded number of errors," *Information Processing Letters*, vol. 127, pp. 27-31, 2017.
[29] Cardoso J., "Business Process Control-Flow Complexity: Metric, Evaluation, and Validation," *International Journal of Web Services Research*, vol. 5, no. 2, pp. 49-76, 2008.
[30] Lim H. W., Kerschbaum F., Wang H., "Workflow Signatures for Business Process Compliance," *IEEE Transactions on Dependable and Secure Computing*, vol. 9, no. 5, pp. 756-769, 2012.