Checking Wrong Decision and Wrong Pattern by Using A Graph-based Method

Kelly Rossa Sungkono
Department of Informatics
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
kelly@its.ac.id

Erina Oktavia Putri
Department of Mathematics
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
erinaoktaviaputri@gmail.com

Habibatul Azkiyah
Department of Mathematics
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
hazkiyah1@gmail.com

Riyanarto Sarno
Department of Informatics
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
riyanarto@if.its.ac.id

***Abstract*— Companies in the world have been considered fraud as a crucial problem. The fraud can be caused by many things, including data manipulation and anomalies in business processes or standard operational procedures (SOP). Anomalies have several types; two of them are a wrong decision and a wrong pattern. A wrong decision, which is not in accordance with the standard list in SOP, occurs as a result of making wrong decisions. On the other hand, the wrong sequence of activities is called a wrong pattern. For detecting those two anomalies, this paper proposed a graph-based method. The graph-based method creates rules for detecting a wrong pattern by measuring the similarity between traces of SOP and the process and checks the attributes of activities to detect a wrong decision. The evaluation uses an event log of the credit application process in a bank. Based on the evaluation, the proposed graph-based method gains 100% for the accuracy value in checking wrong pattern and wrong decision.**

Keywords*— *fraud, wrong decision detection, wrong pattern detection

I. INTRODUCTION

So far, fraud has become a severe problem that is difficult to handle in various sectors, including the banking sector. It is stated that the gross organization income is decreasing around 5% every year [1]. Because no system is perfect and without loopholes, the business of making a secure system is always something that is constantly being developed. Thus, given these statements, identifying fraud by the anomaly detection algorithm is a must.

There are several anomaly detection algorithms; one of them is process mining. Process mining forms a process model containing sequence or parallel relationships, i.e., XOR, AND, and OR [2] based on the event log [3]–[6]. Analyzing the process model based on the logs provides many benefits for various aspects, including fraud detection [7].

This paper proposed a graph-based algorithm as the new process mining method to identify two types of anomalies. Those types are a wrong pattern and a wrong decision because this paper focuses on detecting anomalies related to activities and relationships. The graph-based method creates rules for detecting a wrong pattern by measuring the similarity between traces of SOP and the process and checks the attributes of activities to detect a wrong decision. The proposed algorithm is built-in Neo4j, a graph-database application.

The proposed graph-based algorithm will be evaluated with an event log and SOP of the credit application process in a bank. The results of the proposed algorithm will be equipped with an accuracy calculation based on a confusion matrix.

II. RELATED LITERATURE

A. Fraud

Definition of fraud is the abuse of organizational systems [8]. When fraud occurs in a business process, the complete SOP (Standard Operating Procedure) can be used to overcome it. Things that are not following the SOP can be categorized as fraud [9]. In order to deal with frauds, anomaly detection algorithms must be presented. In short, an anomaly occurs when an error occurs in the component log. It can be an event that is not registered in the logs or an event that is registered in a duplicate [10]–[12]. In the business processes, there are six types of anomalous attributes or fraudulent behavior [9], [13]. But, this paper will only focus on the two of them, which are wrong decision and wrong pattern. As the name suggests, wrong decision is a type of anomaly that occurs as a result of making wrong decisions and not following the standards listed in standard operating procedures (SOP). Meanwhile, wrong pattern is a type of anomaly that occurs due to the wrong sequence of activities, which is not following the sequence of activities stated in standard business processes [9].

B. Bank Credit Application Process

SOP of credit application process in a bank is showed in Fig. 1. The first activity carried out is receive applications, namely receiving credit applications. The data received is in the form of statutory and regulatory documents required for credit applications.

Check for completeness is an activity to check the completeness of the legal and regulatory document provided by the creditor. If the document is incomplete, the get more info activity will be executed to make sure the creditor knows what to do to complete the document. This activity can be repeated until the document is totally completed. Once done, the next process is to check the loan amount.

Check loan amount is the activity of checking the amount of money to be loaned to the creditor. After that, system will perform checks for loan amount if the loan amount is more than 500. However, system will perform checks for small amount if the loan amount is less than or equal to 500. If the loan amount is more than 500 but the next activity is "perform checks for small amount", then the process is wrong decision, and vice versa. Then, the make decision is executed. It will decide whether the creditors get their request approved or not. If the credit application is rejected, system will notify rejection and the finish application activity can be executed.

978-1-6654-4059-2/21/$31.00 ©2021 IEEE

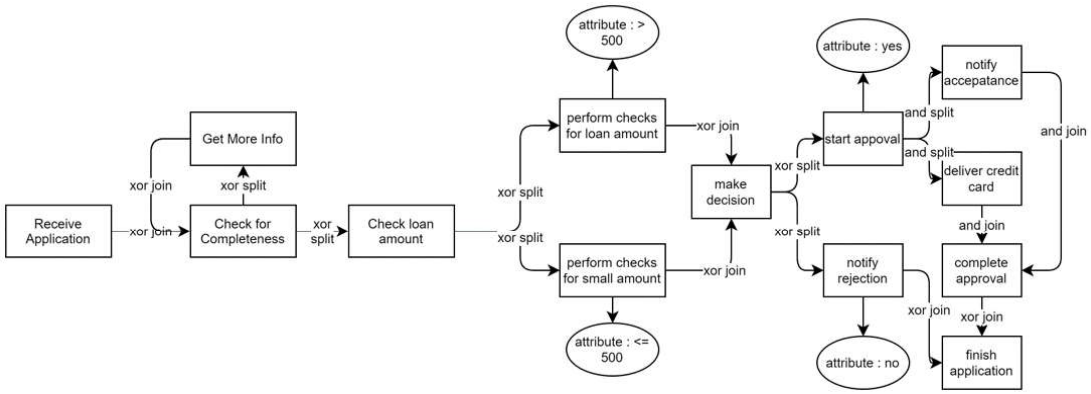

Fig. 1. SOP of Credit Application Process in a Bank

If the creditor does not get any rejection notification, then the start approval activity is executed. System will notify the acceptance and the creditor will deliver the credit card to complete the approval. Finally, the creditor can finish the application.

As previously explained, SOP can be used as a reference for detecting fraud. In this paper, check wrong decision can be done with analyzing SOP and the event log. Note that in SOP, there is always a condition that must be required in order to proceed to the next step. Then, it can be concluded that it is a wrong decision if the attribute of event log does not meet the requirements in SOP. Meanwhile, checking for wrong pattern in this process can be done by analyzing the similarity between SOP graph and event log graph.

C. Jaccard Similarity

Jaccard Similarity or *Jaccard Coefficient* is one of several methods used to measure similarities between sets. The larger the value of the *Jaccard Coefficient*, the higher the similarity of the sample [14]. In some cases, *Jaccard Similarity* can be used to calculate word similarities, detect paperwork plagiarism, and to diagnose computer damage. In this research, the *Jaccard Similarity* is used to find the wrong pattern in a business process [15]. The mathematical representation of the *Jaccard Similarity* is shown in Eq. (1). Meanwhile, the *Jaccard Similarity* algorithm in Neo4j is *gds.alpha.similarity.jaccard(A, B).*

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} \quad (1)$$

TABLE I. EVENT LOG

Case_ID	Activity	Time	Attributes
PP1	Receive Application	3/8/16 10:32 AM	null
PP1	Check for Completeness	3/8/16 10:35 AM	null
PP1	Get More Info	3/8/16 10:50 AM	null
PP1	Check Loan Amount	3/8/16 10:53 AM	650
PP1	Perform Checks for Loan Amount	3/8/16 11:08 AM	null
PP1	Make Decision	3/8/16 11:11 AM	No
- -	- -	- -	- -
PP4	Finish Application	3/8/16 3:41 PM	null

TABLE II. THE QUERY MODELING EVENT LOG INTO GRAPH-MODEL

No	Code
	Load Event Log File
1	LOAD CSV with headers FROM "file:///LoanProcess.csv" AS line Merge (:Activity{CaseId:line.Case_ID, Name:line.Activity, Time:line.Time, Attributes:line.Attributes})
2	LOAD CSV with headers FROM "file:///LoanProcess.csv" AS line Merge (:CaseActivity {Name:line.Activity })
	Create SEQUENCE Relationships
3	MATCH (c:Activity) WITH COLLECT(c) AS Caselist UNWIND RANGE(0,Size(Caselist) - 2) as idx WITH Caselist[idx] AS s1, Caselist[idx+1] AS s2 MATCH (b:CaseActivity),(a:CaseActivity) WHERE s1.CaseId = s2.CaseId AND s1.Name = a.Name AND s2.Name = b.Name MERGE (a)-[r:SEQUENCE]->(b)
	Create XOR Relationships
4	MATCH (b)-[r]->(a) WHERE size((b)-->())>1 AND size((a)<--())=1 AND (size((a)-->())=1 OR size((a)-->())>1) CREATE (b)-[:XORSPLIT]->(a) DELETE r
5	MATCH (b)-[r]->(a) WHERE (size((b)-->())=1 OR size((b)-->())>1) AND size((a)<--())>1 CREATE (b)-[:XORJOIN]->(a) DELETE r
	Create AND Relationships
6	MATCH (a)<-[r]-(b)-[s]->(c) WHERE size((b)-->())>1 AND size((c)-->())=size((b)-->()) AND size((a)-->())=size((b)-->()) AND not (a)-[:SEQUENCE]->(b) AND not (c)-[:SEQUENCE]->(b) MERGE (a)<-[:ANDSPLIT]-(b)-[:ANDSPLIT]->(c) DELETE r,s
7	MATCH (a)-[r]->(b)<-[s]-(c) WHERE size((b)<--())>1 AND size((c)-->())=size((b)<--()) AND size((a)-->())=size((b)<--()) AND not ()-[:ANDSPLIT]->(b) MERGE (a)-[:ANDJOIN]->(b)<-[:ANDJOIN]-(c) DELETE r,s
	Create Invisible Task
8	MATCH (c)<-[r]-(b)-[s]->(a) WHERE (TYPE(s)='XORSPLIT' AND TYPE(r)='XORJOIN') CREATE (i:Invisible_Task {Name: "Inv_Task"}) WITH b, i, c, r, s CALL apoc.create.relationship(b, type(s), {}, i) YIELD rel as relation1 CALL apoc.create.relationship(i, type(r), {}, c) YIELD rel as relation2 DELETE r

TABLE III. THE QUERY TO CREATE 12 PROCESS MODEL

Create Node Activity and Relationships
CREATE (ra:SOP_ProcessX {Name:'Receive Application'}), (cfc:SOP_ProcessX {Name:'Check for Completeness'}), (gmi:SOP_ProcessX {Name:'Get More Info'}), (cla:SOP_ProcessX {Name:'Check Loan Amount'}), (pcfla:SOP_ProcessX {Name:'Perform Checks for Loan Amount'}), (pcfsa:SOP_ProcessX {Name:'Perform Checks for Small Amount'}), (md:SOP_ProcessX {Name:'Make Decision'}), (sa:SOP_ProcessX {Name:'Start Approval'}), (dcc:SOP_ProcessX {Name:'Deliver Credit Card'}), (na:SOP_ProcessX {Name:'Notify Acceptance'}), (nr:SOP_ProcessX {Name:'Notify Rejection'}), (ca:SOP_ProcessX {Name:'Complete Approval'}), (fa:SOP_ProcessX {Name:'Finish Application'})
1. Relationships in SOP Process 1 CREATE (ra)-[:XORJOIN]->(cfc)-[:XORSPLIT]->(gmi)-[:XORJOIN]->(cfc)-[:XORSPLIT]->(cla)-[:XORSPLIT]->(pcfla)-[:XORJOIN]->(md)-[:XORSPLIT]->(sa)-[:ANDSPLIT]->(na)-[:ANDJOIN]->(ca)-[:XORJOIN]->(fa) **2. Relationships in SOP Process 2** CREATE (ra)-[:XORJOIN]->(cfc)-[:XORSPLIT]->(gmi)-[:XORJOIN]->(cfc)-[:XORSPLIT]->(cla)-[:XORSPLIT]->(pcfla)-[:XORJOIN]->(md)-[:XORSPLIT]->(sa)-[:ANDSPLIT]->(dcc)-[:ANDJOIN]->(ca)-[:XORJOIN]->(fa) **3. Relationships in SOP Process 3** CREATE (ra)-[:XORJOIN]->(cfc)-[:XORSPLIT]->(gmi)-[:XORJOIN]->(cfc)-[:XORSPLIT]->(cla)-[:XORSPLIT]->(pcfla)-[:XORJOIN]->(md)-[:XORSPLIT]->(nr)-[:XORJOIN]->(fa) **4. Relationships in SOP Process 4** CREATE (ra)-[:XORJOIN]->(cfc)-[:XORSPLIT]->(gmi)-[:XORJOIN]->(cfc)-[:XORSPLIT]->(cla)-[:XORSPLIT]->(pcfsa)-[:XORJOIN]->(md)-[:XORSPLIT]->(sa)-[:ANDSPLIT]->(na)-[:ANDJOIN]->(ca)-[:XORJOIN]->(fa) **5. Relationships in SOP Process 5** CREATE (ra)-[:XORJOIN]->(cfc)-[:XORSPLIT]->(gmi)-[:XORJOIN]->(cfc)-[:XORSPLIT]->(cla)-[:XORSPLIT]->(pcfsa)-[:XORJOIN]->(md)-[:XORSPLIT]->(sa)-[:ANDSPLIT]->(dcc)-[:ANDJOIN]->(ca)-[:XORJOIN]->(fa) **6. Relationships in SOP Process 6** CREATE (ra)-[:XORJOIN]->(cfc)-[:XORSPLIT]->(gmi)-[:XORJOIN]->(cfc)-[:XORSPLIT]->(cla)-[:XORSPLIT]->(pcfsa)-[:XORJOIN]->(md)-[:XORSPLIT]->(nr)-[:XORJOIN]->(fa) **7. Relationships in SOP Process 7** CREATE (ra)-[:XORJOIN]->(cfc)-[:XORSPLIT]->(cla)-[:XORSPLIT]->(pcfla)-[:XORJOIN]->(md)-[:XORSPLIT]->(sa)-[:ANDSPLIT]->(na)-[:ANDJOIN]->(ca)-[:XORJOIN]->(fa) **8. Relationships in SOP Process 8** CREATE (ra)-[:XORJOIN]->(cfc)-[:XORSPLIT]->(cla)-[:XORSPLIT]->(pcfla)-[:XORJOIN]->(md)-[:XORSPLIT]->(sa)-[:ANDSPLIT]->(dcc)-[:ANDJOIN]->(ca)-[:XORJOIN]->(fa) **9. Relationships in SOP Process 9** CREATE (ra)-[:XORJOIN]->(cfc)-[:XORSPLIT]->(cla)-[:XORSPLIT]->(pcfla)-[:XORJOIN]->(md)-[:XORSPLIT]->(nr)-[:XORJOIN]->(fa) **10. Relationships in SOP Process 10** CREATE (ra)-[:XORJOIN]->(cfc)-[:XORSPLIT]->(cla)-[:XORSPLIT]->(pcfsa)-[:XORJOIN]->(md)-[:XORSPLIT]->(sa)-[:ANDSPLIT]->(na)-[:ANDJOIN]->(ca)-[:XORJOIN]->(fa) **11. Relationships in SOP Process 11** CREATE (ra)-[:XORJOIN]->(cfc)-[:XORSPLIT]->(cla)-[:XORSPLIT]->(pcfsa)-[:XORJOIN]->(md)-[:XORSPLIT]->(sa)-[:ANDSPLIT]->(dcc)-[:ANDJOIN]->(ca)-[:XORJOIN]->(fa) **12. Relationships in SOP Process 12** CREATE

Create Node Activity and Relationships
(ra)-[:XORJOIN]->(cfc)-[:XORSPLIT]->(cla)-[:XORSPLIT]->(pcfsa)-[:XORJOIN]->(md)-[:XORSPLIT]->(nr)-[:XORJOIN]->(fa)

TABLE IV. CYPER FOR CHECKING WRONG DECISION

No	Checking Wrong Decision
1	MATCH (a:Activity) WITH collect(a) AS Caselist UNWIND RANGE(0, Size(Caselist)-2) AS idx WITH Caselist[idx] AS s1, Caselist[idx+1] AS s2 MATCH (b:Activity {Name:'Check Loan Amount'}) WHERE s1.Name = b.Name AND s1.CaseId = b.CaseId AND s1.Attributes > 500 AND s2.Name = 'Perform Checks for Small Amount' OR s1.Name = b.Name AND s1.CaseId = b.CaseId AND s1.Attributes <= 500 AND s2.Name = 'Perform Checks for Loan Amount' SET s2.Information = 'It is a wrong decision' RETURN s2.CaseId as Case_Id, s1.Attributes as Atribute , s1.Name as From, s2.Name as To, s2.Information as Result
2	MATCH (a:Activity) WITH collect(a) AS Caselist UNWIND RANGE(0, Size(Caselist)-2) AS idx WITH Caselist[idx] AS s1, Caselist[idx+1] AS s2 MATCH (b:Activity {Name:'Make Decision'}) WHERE s1.Name = b.Name AND s1.CaseId = b.CaseId AND s1.Attributes = 'Yes' AND s2.Name = 'Notify Rejection' OR s1.Name = b.Name AND s1.CaseId = b.CaseId AND s1.Attributes = 'No' AND s2.Name = 'Start Approval' SET s2.Information = 'It is a wrong decision' RETURN s2.CaseId as Case_Id, s1.Attributes as Attribute, s1.Name as From, s2.Name as To, s2.Information as Result

TABLE V. THE QUERY FOR ADDING "NO" PROPERTY

MATCH	SET
(a {Name:'Receive Application'}),	a.no=1,
(b {Name:'Check for Completeness'}),	b.no=2,
(c {Name:'Get More Info'}),	c.no=3,
(d {Name:'Check Loan Amount'}),	d.no=4,
(e {Name:'Perform Checks for Loan Amount'}),	e.no=5,
(f {Name:'Perform Checks for Small Amount'}),	f.no=6,
(g {Name:'Make Decision'}),	g.no=7,
(h {Name:'Start Approval'}),	h.no=8,
(i {Name:'Notify Rejection'}),	i.no=9,
(j {Name:'Notify Acceptance'}),	j.no=10,
(k {Name:'Deliver Credit Card'}),	k.no=11,
(l {Name:'Complete Approval'}),	l.no=12,
(m {Name:'Finish Application'})	m.no=13

TABLE VI. CYPER FOR CHECKING WRONG PATTERN

Pseudocode	Code

count <- 0
for every process in SOP:
  for every trace in event log:
  similarity <- method of checking similarity(trace,process)
  if count == 0 :
  if similarity > 0.0:
  count <- 1
if count == 0:
  trace is wrong pattern
	MATCH (a:Activity {Trace})
	WITH collect(a.no) as No_activity
	MATCH (b:SOP)
	WITH No_activity, b,collect(b.no) as No_ActivitySOP
	RETURN No_activity, No_ActivitySOP, gds.alpha.similarity.jaccard(No_activity,No_ActivitySOP) as similarity

III. RESEARCH MODEL

A. Change SOP into Graph Model

The next step is to change Standard Operational Procedure (SOP) into graph-model by using Neo4j. In the bank credit application process there are 13 activities and 4 relations i.e., XORSPLIT, XORJOIN, ANDSPLIT, and ANDJOIN. The SOP is divided into 12 sequential process model i.e., 6 process models with iteration and 6 others

without iteration. The 12 process models will be used for matching the process model in event log with the process model in SOP. The query structure to create 12 process model shown in TABLE III. SOP_Process(X) in this table means the SOP Process number X where X-value numbers 1 to 12. For example, SOP_Process1, SOP_Process2, until the last SOP_Process12.

B. Checking Wrong Decision

In SOP, there are several attributes that determine what activities will be carried out after doing the previous activity. There are 2 activities, namely "Check Loan Amount" and "Make Decision".

In the "Check Loan Amount" activity, there is an attribute in the form of the loan amount that can determine the activity to be carried out next. If the loan amount is more than 500, the next activity is "Perform Checks for Loan Amount", if the loan amount is less than equal to 500 then the next activity is "Perform Checks for Small Amount". If the two conditions occur in reverse in the implementation, in this case it is called a wrong decision.

The second activity is "Make Decision". There are attributes in the form of strings "Yes" and "No". If the attribute is "Yes" then the next activity is "Start Approval", if the attribute is "No" then the next activity is "Notify Rejection". Just like the previous case, if the opposite occurs it is called a wrong decision. To find all wrong decision in business process model, the cypher shown in TABLE IV.

C. Checking Wrong Pattern

To find a wrong pattern in the business process model, this research uses the Jaccard similarity algorithm, which is available in Neo4j. The step taken is adding a new property in the form of "no" from numbers 1 to 13 on all activity nodes. Here is the description of each “no” property: (1) Receive Application (2) Check for Completeness (3) Get More Info (4) Check Loan Amount (5) Perform Checks for Loan Amount (6) Perform Checks for Small Amount (7) Make Decision (8) Start Approval (9) Notify Rejection (10) Notify Acceptance (11) Deliver Credit Card (12) Complete Approval (13) Finish Application. The detail of query to add property "no" is shown in TABLE V. After adding "no" properties, the next step is to make a match and collect "no" property from each activity of the model process in the event log with the process model in the SOP. After that, calculate the similarity of activity in event log process models with 12 SOP process models. The pseudocode to find wrong pattern in business process shown in TABLE VI. The first step is matching all activities of trace or CaseId on eventlog, for example MATCH (a: Activity {CaseId: PP1}). Next, match activities in SOP_ProcessX. Then, the similarity can be calculated by Jaccard Similarity algorithm.

IV. Result and Discussion

A. Graph Model

The results of modeling the event log in TABLE I into a graph that have been carried out using the algorithm in the TABLE II can be seen in Fig. 2. Meanwhile, the graph of the SOP is shown in Fig. 3 to Fig. 14. Fig. 3 to Fig. 8 is a process model of the SOP by looping the activity “Check for Complete” to “Get More Info” then returning to “Check for Complete”. Meanwhile, Fig. 9 to Fig. 14 are the process models of the SOP without iteration.

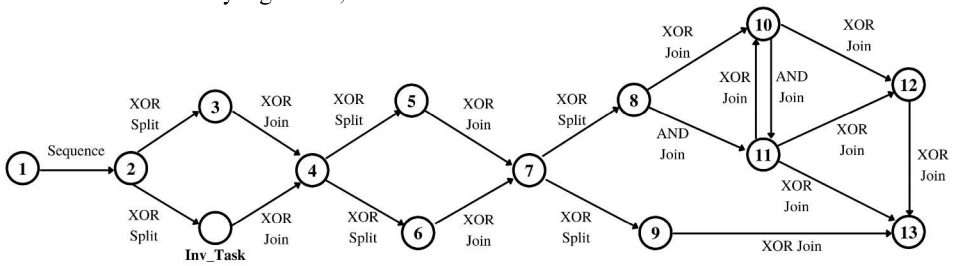

Fig. 2. Event Log Graph-Model

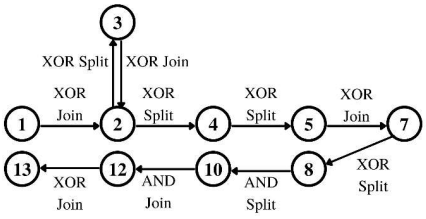

Fig. 3. SOP Process 1

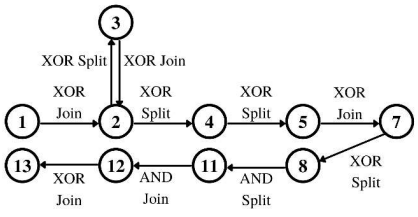

Fig. 4. SOP Process 2

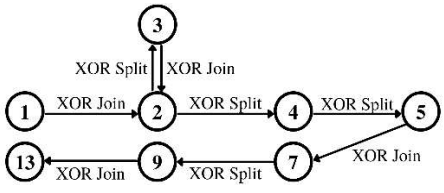

Fig. 5. SOP Process 3

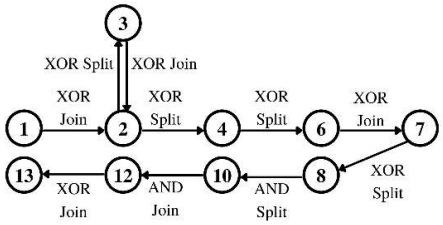

Fig. 6. SOP Process 4

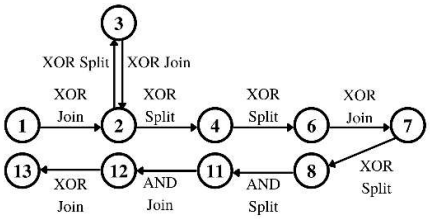

Fig. 7. SOP Process 5

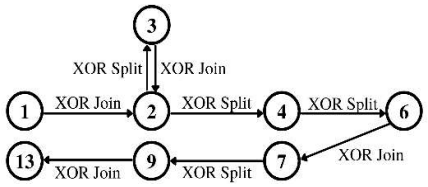

Fig. 8. SOP Process 6

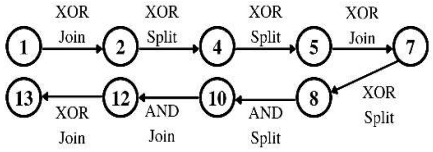

Fig. 9. SOP Process 7

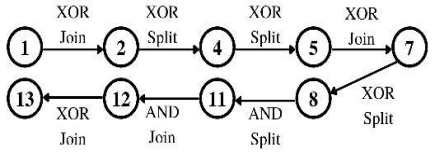

Fig. 10. SOP Process 8

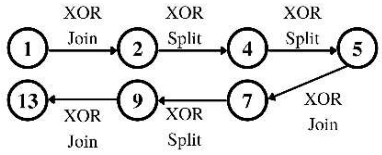

Fig. 11. SOP Process 9

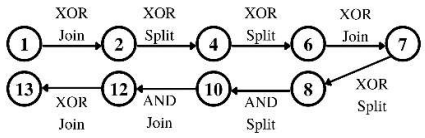

Fig. 12. SOP Process 10

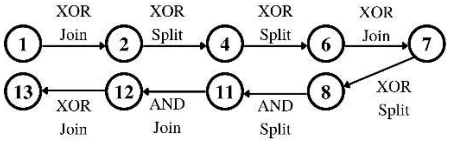

Fig. 13. SOP Process 11

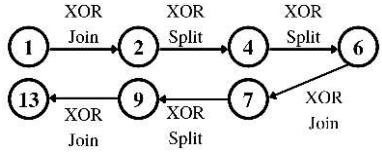

Fig. 14. SOP Process 12

TABLE VII. WRONG DECISION RESULT

Case_Id	Attribute	From	To	Result
PP2	1000	Check Loan Amount	Perform Checks for Small Amount	"It is a wrong decision"

TABLE VIII. SIMILARITY RESULT

Case ID	Similar with
PP1	SOP Process 3, SOP Process 9
PP2	SOP Process 10, SOP Process 11
PP3	SOP Process 10, SOP Process 11
PP4	Not similar with all SOP processes

B. Wrong Decision

In the TABLE IV, the algorithm for finding wrong decisions in the money loan business process model has been explained. From all the matches, the result is shown in the TABLE VIII.

Based on TABLE VII, the "Check Loan Amount" activity on Case ID PP2 has the Attribute of 1000 USD. If according to the SOP, the next activity to be carried out is "Perform Checks for Small Amount" because the loan amount is more than 500 USD. Therefore, PP2 is a wrong decision caused by wrong activity after "Check Loan Amount".

C. Wrong Pattern

Based on the algorithms mentioned above, checks are carried out and the results are listed in TABLE VIII. In Case PP1 which is compared with SOP Process 1, there is a similarity with a value of 0.0. This means that there is an activity on PP1 which is not the same as the activity in the SOP. Then in the comparison of PP1 with SOP Process 3, all the similarity values are the same and there is no 0.0. This means that all activities on PP1 are similar to those in SOP Process 3. Thus, PP1 is not a wrong pattern. In the same way, it can be concluded that PP2 and PP3 are also not the wrong pattern. Meanwhile, from all PP4 checks with each SOP process, the similarity value that appears is always 0.0. It means that PP4 is the wrong pattern.

D. Accuracy Calculation

Based on the results obtained, the accuracy of the algorithm will be calculated using a confusion matrix. As in the previous explanation, the first step is to determine True Positive (TP), False Positive (FP), True Negative (TN), and False Negative (FN). Then the accuracy can be calculated with the formula (TP+TN)/(TP+FP+FN+TN). If the value of accuracy is close to 1, then the algorithm is nearly perfect.

1) Calculate Accuracy of Wrong Decision

Based on the wrong decision algorithm results, a comparison of the predicted results with the actual results in TABLE IX can be shown. So it can be determined 1 TP, 0 FP, 0 TN, and 3 FN.

$$Accuracy = \frac{1+3}{1+0+0+3} \times 100\% = 100\% \quad (3)$$

2) Calculate Accuracy of Wrong Pattern

Based on the wrong pattern algorithm results, a comparison of the predicted results with the actual results in TABLE X can be shown. So it can be determined 1 TP, 0 FP, 0 TN, and 3 FN.

$$Accuracy = \frac{1+3}{1+0+0+3} \times 100\% = 100\% \quad (4)$$

TABLE IX. COMPARISON OF THE ACTUAL AND PREDICTED RESULTS

Case_ID	Actual	Predicted
PP1	Not wrong decision	Not wrong decision
PP2	Wrong decision	Wrong decision
PP3	Not wrong decision	Not wrong decision
PP4	Not wrong decision	Not wrong decision

TABLE X. COMPARISON OF THE ACTUAL AND PREDICTED RESULTS

Case_ID	Actual	Predicted
PP1	Not wrong pattern	Not wrong pattern
PP2	Not wrong pattern	Not wrong pattern
PP3	Not wrong pattern	Not wrong pattern
PP4	Wrong pattern	Wrong pattern

CONCLUSION

According to the aforementioned experimental results, it can be concluded that process mining using a graph-based method can be used to detect a wrong decision and a wrong pattern in business processes, especially in the credit application process in a bank. Before detecting frauds, the event log and SOP are modelled into graph models. In order to check whether a wrong decision occurs or not, we have to pay attention to SOP. Several attributes determine what activities will be carried out after doing the previous activity. If the conditions occur in reverse in the implementation, in this case, it is called a wrong decision. In this process, PP2 is a wrong decision caused by inappropriate activity after "Check Loan Amount". Meanwhile, to find a wrong pattern in the business process model, this research uses the Jaccard similarity algorithm, available in Neo4j. In this process, it can be concluded that PP4 is a wrong pattern because the similarity value between SOP and event log that appears is always 0.0.

In this research, the algorithms used to find a wrong decision and a wrong pattern have a value of accuracy 100%. However, some queries have to be executed manually to get answers to the problems being researched. So that this method can be maximized with other software to solve this problem; thus, this method needs to be reviewed and retested to obtain more accurate results. Hopefully, in future research, this method can be used and developed to detect other anomalies.

ACKNOWLEDGMENT

This research was funded by the Indonesian Ministry of Education and Culture under Penelitian Terapan Unggulan Perguruan Tinggi (PTUPT) Program managed by Institut Teknologi Sepuluh Nopember (ITS) and under Riset Inovatif-Produktif (RISPRO) Invitation Program managed by Lembaga Pengelola Dana Pendidikan (LPDP).

REFERENCES

[1] A. Ramírez-Orellana, M. J. Martínez-Romero, and T. Mariño-Garrido, "Measuring fraud and earnings management by a case of study: Evidence from an international family business," *Europen Journal of Family Business*, vol. 7, no. 1–2, pp. 41–53, 2017, doi: 10.1016/j.ejfb.2017.10.001.

[2] R. Sarno and K. R. Sungkono, "Hidden markov model for process mining of parallel business processes," *International Review of Computer and Software*, vol. 11, no. 4, pp. 290–300, 2016, doi: 10.15866/irecos.v11i4.8700.

[3] R. Samo, R. D. Dewandono, T. Ahmad, M. F. Naufal, and F. Sinaga, "Hybrid association rule learning and process mining for fraud detection," *IAENG International Journal of Computer Science*, vol. 42, no. 2, pp. 59–72, 2015.

[4] A. Cahyapratama, K. R. Sungkono, and R. Sarno, "Gap analysis business process model by using structural similarity," *Indonesian Journal of Electrical Engineering Computer Science*, vol. 18, no. 1, pp. 124–134, 2019, doi: 10.11591/ijeecs.v18.i1.pp124-134.

[5] R. Sarno and K. R. Sungkono, "A survey of graph-based algorithms for discovering business processes," *International Journal of Advances in Intelligent Informatics*, vol. 5, no. 2, pp. 137–149, 2019, doi: 10.26555/ijain.v5i2.296.

[6] K. R. Sungkono and R. Sarno, "Patterns of fraud detection using coupled Hidden Markov Model," *Proceeding - 2017 3rd International Conference on Science in Information Technology: Theory and Application of IT for Education, Industry and Society in Big Data Era, ICSITech 2017*, vol. 2018-January, pp. 235–240, 2017, doi: 10.1109/ICSITech.2017.8257117.

[7] O. Allani and S. A. Ghannouchi, "Verification of BPMN 2.0 Process Models: An Event Log-based Approach," *Procedia Computer Science*, vol. 100, pp. 1064–1070, 2016, doi: 10.1016/j.procs.2016.09.282.

[8] F. T. Dezoort and P. D. Harrison, "Understanding Auditors' Sense of Responsibility for Detecting Fraud Within Organizations," *Journal of Business Ethics*, pp. 857–874, 2018, doi: 10.1007/s10551-016-3064-3.

[9] R. Sarno, F. Sinaga, and K. R. Sungkono, "Anomaly detection in business processes using process mining and fuzzy association rule learning," *Journal of Big Data*, vol. 7, no. 1, pp. 1–19, 2020, doi: 10.1186/s40537-019-0277-1.

[10] G. Baader and H. Krcmar, "Reducing false positives in fraud detection: Combining the red flag approach with process mining," *International Journal of Accounting Information Systems*, vol. 31, no. February, pp. 1–16, 2018, doi: 10.1016/j.accinf.2018.03.004.

[11] A. Mandal, A. Nandi, S. Atreja, G. B. Dasgupta, and S. Bhattacharya, "Anomaly detection using program control flow graph mining from execution logs," in *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016. doi: 10.1145/2939672.2939712.

[12] H. Darmawan, R. Sarno, A. S. Ahmadiyah, K. R. Sungkono, and C. S. Wahyuni, "Anomaly detection based on control-flow pattern of Parallel Business Processes," *Telkomnika (Telecommunication Computing Electronics and Control*, vol. 16, no. 6, pp. 2809–2816, 2018, doi: 10.12928/TELKOMNIKA.v16i6.10568.

[13] R. Sarno and F. P. Sinaga, "Business process anomaly detection using ontology-based process modelling and Multi-Level Class Association Rule Learning," in *Proceeding - 2015 International Conference on Computer, Control, Informatics and Its Applications: Emerging Trends in the Era of Internet of Things, IC3INA 2015*, Jan. 2016, pp. 12–17, doi: 10.1109/IC3INA.2015.7377738.

[14] D. Zhang, X. You, S. Liu, and K. Yang, "Multi-Colony Ant Colony Optimization Based on Generalized Jaccard Similarity Recommendation Strategy," *IEEE Access*, vol. 7, pp. 157303–157317, 2019, doi: 10.1109/ACCESS.2019.2949860.

[15] "Jaccard Similarity - Neo4j Graph Data Science." https://neo4j.com/docs/graph-data-science/current/alpha-algorithms/jaccard/ (accessed Jun. 01, 2021).