Determining Linear Temporal Logic Formula for Decomposed Process Model

Maryamah
Institute Technology Sepuluh Nopember
Surabaya, Indonesia
maryamahfaisol02@gmail.com

Riyanarto Sarno
Institute Technology Sepuluh Nopember
Surabaya, Indonesia
riyanarto@if.its.ac.id

Afina Lina Nurlaili
Institute Technology Sepuluh Nopember
Surabaya, Indonesia
afina.lina12@gmail.com

***Abstract*—Process discovery is a process to observe behaviour in the event log and to build a model for the next process. In addition, it is an important process because its strength to predict the time, and cost. After constructing the model, the model has to be set into some parts by using decomposed process algorithm. So that, the result will be easier to be analyzed. A decomposed process can implementation inductive miner algorithm. However, decomposed using inductive miner have limited relation in process. To overcome this problem, this paper proposed decomposed model by using Linear Temporal Logic (LTL) to find the rule and build the process model automatically without constructing from the first step and also have many notations to formalize relation of activity. In addition, LTL is a method to build the rule and check the workflow of the process logs whether the logs have the parallel process. So that, by using the proposed method LTL will be used for getting a process model in less time with average time 1 second and more accurate result.**

Keywords—; Discovery; Process Mining; Decomposed; Linear Temporal Logic (LTL).

I. INTRODUCTION

An event log is a record from the process that contains activity case on sequence data and gets from executed in information system [1]**.** In addition, it is an important process because its strength to predict the time, cost, and used the resource. Knowing the benefits of event logs it is necessary to understand the behaviour in the event logs so that they can be model to get a better looking for analysis. The process of behavior observed in the event log and constructing a model is named as discovery.

Discovery is an important process in process mining [2]. In addition, discovery becomes a challenging process because it is an automatic model which construct current activities and the variations [3]. Nowadays, discovery process gains much attention among researcher and practitioner to get a good result [4]. The result is influenced by the quality of event log. Sometimes, event logs have noise which can influence the activity of discovery process in the event log and the model which will be constructed.

The completion of the process discovery can use many kinds of algorithms such as Alpha, Alpha +, Alpha ++, Heuristic miner, Genetic miner, and Inductive miner algorithms. Some of this algorithms have their respective advantages. One of a commonly used algorithm is an inductive miner. Inductive miner has advantages in filtering noise on input parameters which will be used as path and produce a model of event log then. There are many methods to a model event log, namely Petri net, UML activity diagram, and Tree. By using inductive miner algorithm, the result of a model will be represented as a tree. Modeling using tree is easy to build and give a specific result (node child is a parallel activity and parent is a sequential activity).

By using inductive miner, operators of the event log can be detected. These operators will be represented into a parallel node in tree model and the sequence of the process will be the leaf of the tree. There are two operators that will be detected, namely "AND" and "XOR". All two operators have a significant difference in conditions, "AND" is a condition from two or more start at the same time, "XOR" is a condition where the activity has multiple choice and must be chosen one of them [3]. After detecting the operator in the event log, the model tree will be generated then.

By generating from the model, tree rule can be defined using decomposed process. The decomposed process is a step of splitting activity that existed in the model into sub-activity. The decomposed algorithm will be used using LTL (Linear Temporal Logic). Decomposed using LTL is done by making the rule according to the conditions of activity both sequential and parallel. So LTL rule will get them through model process and rule can be used to build a new model if there is new event log data which want to be processed without going through process discovery. Inductive miner can produce operator and decompose from process model input but inductive miner algorithm have limited relation in process. To overcome this problem, this paper proposed decomposed model by using LTL (Linear Temporal Logic) to find the rule and build the process model automatically without constructing from the first step and also have many notations to formalize relation of activity. In addition, LTL is a method to build the rule and check the workflow of the process logs whether the logs have the parallel process**.** So that, by using the proposed method, LTL can be used for getting a process model in less time and automatically process model without doing process discovery so it can have a more accurate result.

II. RELATED WORK

There is a much-decomposed algorithm in previous research [5-10] for two main problems namely process discovery and replay (Conformance checking). Replay (Conformance checking) is diagnosed the difference between behavior in event log and model. The solution from the main problem is resolving the genetic framework to discovery and replay using ILP miner. Discovery process begins with observe behavior in the event log and discover activity cluster from event log than build graph. Split event log and discover and also merge. For replay begin from event log and model Petri net process at same time. In event log discover matrix (from process discovery), in Petri net create matrix than create a graph. With graph model can find a cluster and merge cluster. Split event log and model with Petri net and put in an array. In model determine activity cost, check both of them and merge and will get log alignment from this process. The result of this paper is discovery process using ILP based decomposed much faster than regular discovery program and replay using ILP based much faster but less accurate [5].

Generic divide-and-conquer approach reveals the core requirement for decomposing problem discovery and conformance checking problem. First, divide the event log to smaller and distribute in another computer then decomposed the process in multiple computers. The general divide-and-conquer approach in particular representation or strategy it gives the incredible growth of event data and easy to explore also investigate the entire spectrum in detail [6]. There are two strategies for decomposed data, Maximal decomposition, and SESE-based decomposition. The Maximal decomposed process will find a minimal subnet to reduce analyzing complexity and the SESE-based identification important sub process. The decomposition significantly reduces the computation time and SESE-based have a faster process for small log size but the two decompositions have small differences and need more large data to resolve [7,8].

Generic divide and conquer for the decomposed process model. In conformance checking, they decomposed any trace in process model to smaller part overlapping model fragment to fits. For discovery split activity to a collection of part overlap activity set with the relevant event and discover a model fragment and a vertex-cut based on graph partition. All edges sharing a source on the same set to ensure split and join not decomposed. Result for conformance checking is reduced time significantly because the large problem decomposed to a smaller problem and also do the process on multiple computers. The discovery problem they partition of the graph a vertex-cut ensuring balance of the result and also minimize the number of vertexes. The overlap of different activity set is small [9].

Decomposed using vertex-cut for recompose activity in process model graph. After the decomposed they create cluster but the number of clusters is large. To find good cluster two agglomerate hierarchical re-composition to merge cluster and reducing the size of the cluster. Hierarchical re-composition in using coupling for proximity measure and cohesion for distance measure. The highest coupling between cluster than larger cluster will merge. The decomposed process is a focus in merge sub model to overall process model but they improve the process with merge sub model based on activity label using a relation in the activity. Clustering process in event log has the good result they can cluster in few cluster in balance (there aren't cluster with many data than another cluster). Data inside cluster in related data each other and unrelated data in another cluster [10].

III. METHOD

A. Modelling Data

Data modeling is a step of changing the event log data into a model. Data modeling can be called as process discovery. This process is done to get a better view of the input event log data. Process discovery is done by applying Inductive miner algorithm, Alpha, Alpha+, Alpha++, and Heuristic Miner. This paper proposes inductive miner in process discovery. The advantages of inductive miner algorithm are to filter noise on the input parameters that will be used as paths and will produce the model. Inductive miner doesn't work in Petri net model which is used in the process tree. Inductive miner algorithm has three main step to solve process discovery problem. The first step is collecting all activity in the event log and save it to array variable then all of the trace activity will be saved. The second step is checking first and last activity whether the event has the same activity, then split and save it into another array variable. The third step is detecting operator by checking activity one by one after the first activity.

There is a plugin to implement inductive miner algorithm in Prom 6. The plugin can find a solution from event log data by constructing tree model which has been separated based on operator or process from a sequential or parallel operator. In Prom, there is another plugin to implement inductive miner named as the visual inductive miner. Visual inductive miner can see visualization to complete discovery process. This research refers to the implementation of plugin model but is implemented by coding manually.

In the depiction of the tree, there are several things to be figured out. The root of a tree is not a representative activity but using the arrow symbol that signifies the running process. Each parent will represent every activity in the event log. These activities can be divided into two activities, namely the activity sequential and parallel activities (activities that are biased into another path). All activities will be parent sequential and have no child. While the parallel activity will be placed into a child. Parallel activity is indicated by the operator. In the event log, there are three main operators "AND" and "XOR". The three operators have significant differences in conditions of "AND" is a condition from two or more start at the same time. "XOR" is a condition where the activity has multiple choice and must choose one. After detecting the operator in the event log then the model tree will be generated. Algorithm inductive miner is shown in Figure 1.

Algorithm Inductive Miner

1. Split start activity and an end activity with condition all start and an end activity same

2. Split activity by an exclusive choice split, or sequence split, or parallel split, or looping until can't be split again.

Figure 1. Algorithm Inductive Miner

Figure 1 is algorithm inductive miner to solve problem discovery and decomposed until having rule result. But decomposed using inductive miner have limited relation in-process model.

B. LTL (Linear Temporal Logic)

LTL is used to check the workflow of the property logs of processes and parallel condition of linear time to work in linear. There are several things to note in the writing of rules using LTL.

1. Next time.

   It is a rule that indicates that the process is sequential. This process is represented by the _o symbol.

2. Eventually.

   It is a decoding process of the assignment to be given or the conditions of the activities that must be met. Symbol writing using < >.

3. Always:

   By using this operator will ensure the property will hold until the last state. Always using symbol [ ].

4. Until:

   By using for hold process and using symbol _U.

5. AND Operator

   Operator AND is a condition where activity has multiple choice but all of the choices must choose all of them then two activity have started at the same time and have same previous activity. Operator AND have symbol /\.

6. XOR Operator

   Operator XOR is a condition where activity has multiple choice but have to choose one of them. To detect XOR operator must check all of the trace inactivity. If an activity has same previous and next activity has different activity. Operator XOR has symbol \ /.

There is plugin LTL checker in Prom 5 can be used for detecting operator in-process model. The result is a general rule from an operator in the event log but not appropriate with an inputted model. In addition, the general rule can't explain the parallel process. To find a correct rule must create it manually and that is not efficient. If the inputted rule is correct then the output of the LTL checker will figure appropriate model based on its process model. LTL is usually not used for process decomposed, LTL is used to get the rule of the existing model so this method is new in implementation on process decomposed. Decompose process using LTL in this research is used to discover a rule of activity in the tree model. The process will find the rule and can build an automatic model from an inputted event log.

C. Decomposed Model using LTL

Propose method in this paper is a decomposed method using LTL for constructing automatic process model. The decomposed process is a step of splitting activity that existed in the model into sub-activity. The decomposed process has many developmental algorithms described in the review literature and algorithm to be used in this research using LTL (Linear Temporal Logic). Step by step the method has two important process, modeling using inductive miner and decomposed model using LTL. Flowchart of the proposed method is shown in Figure 2.

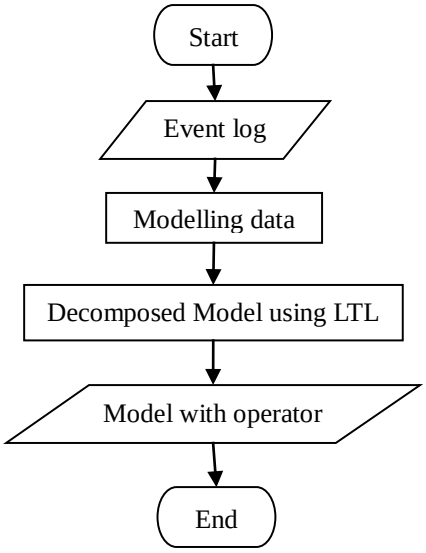

Figure 2. Proposed method

Step by step process in this paper is shown in Figure 1, from discovering process in the data modeling process until have resulted is as follows:

1. Input event log, check start activity and end of an activity. If correct then save in array variable and build a tree.
2. There are three variables that support to detect operators: same, different, and gap. Same is a variable that stores the same activity in a single trace. Different stores different activities and gaps to store parallel activity.
3. Check the first and last activity on all traces, if the same then the activity will be stored on other variables and removed from the activity list. Check the second activity whether each trace is the same. If the same then removed and led to the next activity. If each trace has a different activity then it will not be removed and go to the next activity check. If the next activity is different then go to the next activity to find the same activity. The different activities will be cut until the same activity.

4. AND: conditions that describe when the same activity but differ in sequence. If conditions same = gap and different = gap-1 will be detected "AND".
5. XOR: a condition that describes when the activity is run the same but after the activity will be different. If the condition same = gap and a different = gap will be detected "XOR".
6. After detection operators completed then the model tree will be built in accordance with the activities stored on each operator
7. Decomposed using LTL to generate rule from preciously process and result of the rule.
8. If there are new event log, using a rule of result decomposed with LTL to build model automatic without process modelling data.

IV. RESULT AND ANALYSIS

A. *Dataset*

The dataset used in the experiment to be performed is the standard operation procedure (SOP) data of the "PETI KEMAS" transport. The SOP consists of activities carried out on the container terminal of Surabaya every day. Event log data consists of several trace and case as well as a total activity carried out by 10862 and 83 cases. Example event log data can be seen in Table 1.

Table 1. Event log

Case ID	Activity
4484425	Document_Entry_via_PDE
4484425	Vessel_Berthing_Process
4484425	Discharge_Container
4484425	Bring_Container_to_Yard
4484425	Stack_Container_in_Yard
4484425	Verification_Document_Quarantine
4484425	Create_Job_Order_Document_Quarantine
4484425	Bring_Container_from_Yard_to_Quarantine
4484425	Stack_Container_in_Quarantine_Area
4484425	Check_Goods_Quarantine
4484425	Create_document_KH/KT
4484425	Stack_Container_in_Yard_From_Quarantine
4484425	Verification_Document_Behandle
4484425	Create_document_SPPB
4484425	Create_Job_Order_Document_Delivery
4484425	Truck_in
4484425	Dispatch_WQ_Delivery_to_CHE
4484425	Determine_Container_Type
4484425	Determining_Dry
4484425	Decide_Task_Before_Lift_Container
4484425	Lift_on_Container_Truck
4484425	Truck_Go_To_Gate_Out
4484425	Check_Container_before_Truck_out
4484425	Truck_Out
4485322	Document_Entry_via_PDE
4485322	Vessel_Berthing_Process
4485322	Discharge_Container
4485322	Bring_Container_to_Yard
4485322	Stack_Container_in_Yard
4485322	Verification_Document_Behandle
4485322	Create_Job_Order_Document_Behandle
4485322	Stack_Container_in_Behandle_Area
4485322	Check_Goods_Behandle
4485322	Create_document_LHP
4485322	Bring_Container_from_Yard_to_Behandle
4485322	Stack_Container_in_Yard_From_Behandle
4485322	Create_document_SPPB
4485322	Create_Job_Order_Document_Delivery
4485322	Truck_in
4485322	Dispatch_WQ_Delivery_to_CHE
4485322	Determine_Container_Type
4485322	Determining_Dry
4485322	Decide_Task_Before_Lift_Container
4485322	Lift_on_Container_Truck
4485322	Truck_Go_To_Gate_Out
4485322	Check_Container_before_Truck_out
4485322	Truck_Out

Table 1 is part of data will be processed in this research. This data contained case id and activity. Case id is path one of trace activity to complete the process.

B. *Result*

Implementation of the proposed method in this research has two main processes is modeling data and decomposed model. In modeling data doing an observe behavior from event log and detect the path of a process is sequential or parallel. The first step is to figure a sequential activity to construct in tree model like Figure 3.

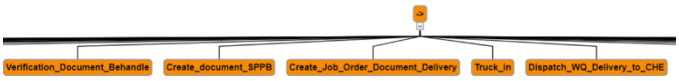

Figure 3. Tree nonparallel process

In Figure 3 we have tree still not have a parallel process. This step first steps to do this research. This step contain save all of the unique activity. Unique activity means process activity contains all activity from all the path. Example the first path has a few process and check the second process if there is a process not contain the first process then will be shown. Next process is processing detection operator in the event log. The operator will detect in this research is AND" and "XOR". After detecting the operator will appear in tree process model like Figure 4.

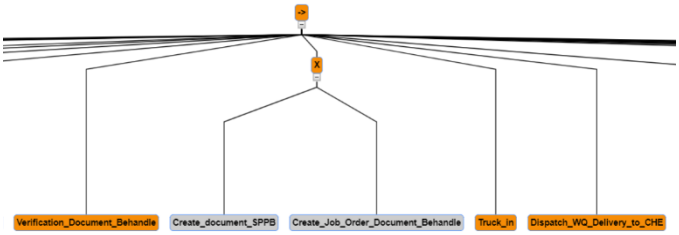

Figure 4. Parallel process

Figure 4 is a parallel tree output from the discovery process. In parallel tree define an operator “X” that means the activity have operator XOR the symbol is construct model based on inductive miner algorithm and LTL rule. Activity in the left side from parent node X is activity have operator parallel. After getting tree model with construct operator parallel then next is process decomposed.

This decomposed process is done by splitting model tree which is gotten from the previous process into a sub model.

The partition of submodel is done based on a characteristic of its model. The division of sequence process is done by taking activities in the model and take the next activity until finding the parallel activity. Otherwise, the sequencing process and the parallel process are saved in the different activities. So that, the decomposed result will be shown differently.

```
run:
Activity start from = Vessel_Berthing_Process and finish = Truck_Out

Trace Activity in event log and size the activity :
8 [[Vessel_Berthing_Process, Discharge_Container, Bring_Container_to_Yard, Stack_Co

Total activity for all trace:
Activity : 32 => {Create_document_SPPB=530, Bring_Container_from_Yard_to_Quarantine

Activity in all trace contain an operator :
Flag : 32 => [0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0,

Operator AND : 1 => [Determining_Dry AND SPLIT [Decide_Task_Before_Lift_Container,
Operator XOR : 2 => [Determine_Container_Type XOR SPLIT [Determining_Dry] XOR JOIN
```

Figure 5. Result LTL

Figure 5 show the result and all step of the proposed method that begins by checking first and last activities. The result can detect the operator that can be seen in figure 5. It shows that there is one operator 'AND' in the event log and shows the detected activity. In addition, there are two operators detected, "XOR" and its activities. By using decomposed LTL, the proposed method can produce a rule that will be used to construct an automatic model.

```
Easy_Document_via_PDE -> _o (Vessel_Berthing_Process)
Vessel_Berthing_Process -> _o (Discard_Yard_planning)
Discard_Yard_planning -> _o (Receive_Container)
Receive_Container -> _o (Stack_Container_in_Yard)
Stack_Container_in_Yard -> _o (Create_Document_Quarantine)
Create_Order_Document_Quarantine -> _o
  (Bring_container_to_Quarantine_Area)
Bring_container_to_Quarantine_Area  -> _o
  (Check_Good_Quarantine)
Verification_Document -> _o  (Create_Document_SPBB)
Create_Job_Order_Document_Behandle -> _o (Truck_in)
Truck_in  -> _o (Dispach_WQ_Delivery_To_Che)
Lift_on_Container_Truck -> _o (Truck_go_to_Gate_out)
Truck_go_to_Gate_out-> _o (Check_container_Before_Truck_out)
Check_container_Before_Truck_out  -> _o (Truck_out)
Determining_Dry-> <> (Decide_Task_Before_Lift_Container /\
  Lift_on_Container_Truck /\ Truck_Go_To_Gate_Out)
Determine_Container_Type-> _o ([Determining_Dry] \/
  [Determining_Uncontainer])
```

Figure 6. Result of rule LTL

Figure 6 show result rule LTL system contain a process consisting of all sequence and parallel activities. The initial LTL Rule shows the previous rule sequence and advanced results from LTL parallel rules of both "AND" and "XOR" operators. By using decomposed LTL, the time execution average is two second. It can be described that decomposed process can run fast. Furthermore, the result of LTL is appropriate with LTL result that is done manually by using LTL checker.

By using the discovery inductive miner process there are drawbacks that the problem of non-free choice can't detect on log events. In that case, it is only a parallel process only.

V. CONCLUSION

By using LTL in the decomposed process, the proposed method can achieve a good result of rule LTL and previous process in modelling process the tree model will split activity using algorithm inductive miner in a few sub model, comparison with decomposed using inductive miner algorithm in prom 6. Sub model contains a sequential or parallel activity. The operator in a parallel model can detect "AND" and "XOR". By using the discovery inductive miner process, there are drawbacks that the problem of non-free choice can't detect on log events. The future research should resolve the non-free choice problem and event log contain double sequential (there are operator inside operator activity) and also detecting "OR" operator.

REFERENCES

[1] D. Rahmawati, M. A. Yaqin and R. Sarno, "Fraud Detection on Event Logs of Goods and Services Procurement Business Process Using Heuristic Miner Algorithm," in *International Conference on Information, Communication Technology and System (ICTS)*, Surabaya, 2016.doi:10.1109/ICTS.2016.7910307

[2] W. M. P. v. d. Aalst, Process Mining - Data Science in Action, 2016.

[3] R. Sarno, Kartini, W. A. Wibowo and A. Solichah, "Time Based Discovery of Parallel Business Processes," in *International Conference on Computer, Control, Informatics and Its Applications*, 2015.doi: 10.1109/IC3INA.2015.7377741

[4] R. Sarno, F. Haryadita, D. Sunaryono and A. Munif, "Model Discovery of Parallel Business Processes using Modified Heuristic Miner," in *International Conference on Science in Information Technology (ICSITech)*, 2015.doi: 10.1109/ICSITech.2015.7407772

[5] H. W. Verbeek and W. v. d. Aalst, "Decomposed Process Mining: The ILP Case," 2014. doi: https://doi.org/10.1007/978-3-319-15895-2_23

[6] v. d. A. Will.M.P, "A General Divide and Conquer Approach for Process Mining," in *Computer Science and Information Systems*, 2013.doi: https://doi.org/10.1093/comjnl/bxx040

[7] W. M. v. d. Aalst, "Decomposing Petri net for Process Mining - A Generic Approach".doi: https://doi.org/10.1007/s10619-013-7127-5

[8] M. d. Leoni, j. Munoz-Gama, J. Cofmona and W. M. V. d. Aalst, "Decomposing on Conformance Checking in Petri Net with Data".

[9] W. M. v. d. Aalst, "Decomposing Process Mining Problem Using Passages".doi: https://doi.org/10.1007/978-3-642-31131-4_5

[10] B. Hompes, H. Verbeek and W. M. v. d. Aalst, "Finding Suitable Activity Cluster for Decomposed Process Discovery". Doi: https://doi.org/10.1007/978-3-319-27243-6_2