Journal of
Information Systems Engineering
and Business Intelligence

Vol.7, No.2, October 2021
Available online at: http://e-journal.unair.ac.id/index.php/JISEBI

Conformance Checking of Dwelling Time Using a Token-based Method

Bambang Jokonowo[1) *], Nenden Siti Fatonah[2)], Emelia Akashah Patah Akhir[3)]

[1)] *Information Systems Department, Faculty of Computer Science, University of Mercu Buana, Indonesia*
Meruya no 1, Jakarta Barat
[1)] bambang.jokonowo@mercubuana.ac.id

[2)] *Informatics Department, Faculty of Computer Science, University of Mercu Buana, Indonesia*
Meruya no 1, Jakarta Barat
[2)] nenden@mercubuana.ac.id

[3)] *Department of Computer Information and Sciences, Universiti Teknologi PETRONAS, Malaysia*
32610, Seri Iskandar, Perak
[3)] emelia.akhir@utp.edu.my

Abstract

Background: Standard operating procedure (SOP) is a series of business activities to achieve organisational goals, with each activity carried to be recorded and stored in the information system together with its location (e.g., SCM, ERP, LMS, CRM). The activity is known as event data and is stored in a database known as an event log.
Objective: Based on the event log, we can calculate the fitness to determine whether the business process SOP is following the actual business process.
Methods: This study obtains the event log from a terminal operating system (TOS), which records the dwelling time at the container port. The conformance checking using token-based replay method calculates fitness by comparing the event log with the process model.
Results: The findings using the Alpha algorithm resulted in the most traversed traces (a, b, n, o, p). The fitness calculation returns 1.0 were produced, missing, and remaining tokens are replied to each of the other traces.
Conclusion: Thus, if the process mining produces a fitness of more than 0.80, this shows that the process model is following the actual business process.

Keywords: Conformance Checking, Dwelling time, Event log, Fitness, Process Discovery, Process Mining

Article history: Received 21 July 2021, first decision: 19 August 2021, accepted 23 September 2021, available online 28 October 2021

I. INTRODUCTION

In the era of big data, almost all data is stored in the information system. Process mining research is currently growing to control flow and monitoring [1]-[3]. Monitoring is done based on event logs stored in the information system [4]. A container port's information system records all transaction processes. The dwelling time is when a container goes down from a docked vessel until it leaves the port. Dwelling time logs are also stored in the information system. Exploring these data using process mining can help measure a port's performance [5]. As such, we can form the process model [6]-[8].We can then do conformance checks to perform audits, as well as compliance checks by replaying the event logs on a process model. Thus, we can measure the performance and visualise the deviations. We can also detect bottlenecks and build predictive models [9], [10].This is necessary because high logistics costs and dwelling time [11] will reduce a port's competitiveness.

The research aims to examine container dwelling time by using process mining to measure a port performance measure. The application of process mining in various case studies have been tested and challenges [10], [12]. However, it has not been tested in the case of container port, where a cross-organisational mining process is at place. Therefore, we perform conformance checking [13] on the dwelling time at a container port. We continue the previous

[*] Corresponding author

ISSN 2443-2555 (online) 2598-6333 (print) © 2021 The Authors. Published by Universitas Airlangga.

doi: http://dx.doi.org/10.20473/jisebi.7.2.129-137

study (i.e., data extraction, discovery) [5] and use one of the process mining techniques, namely conformance checking, which compares the standard operating procedure (SOP) with the process model generated from the event logs.

II. Literature Review

Process mining is research that is engaged between machine learning and data mining and is common in business intelligence environment [14], [15]. Process mining aims to find, monitor and improve the actual process by extracting knowledge from event logs available in the information system. Process mining compare event data (i.e., observed behaviour) and process models (handmade or automatic discovery). Process models can also be used to describe the realignment of what is happening. With the increasing abundance of event data and due to the limitations of handmade models, a process model needs to be created [4], [7], [16]. As such, we can discover the actual processes and the existing process models can be evaluated and enhanced.

Process model is used for [7] verification, to find errors in the system or procedure (e.g., potential deadlocks, bottlenecks); and performance analysis, to understand the multiple factors that affect response times, service levels, etc.

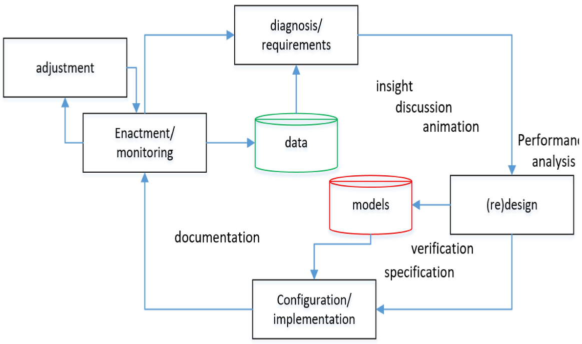

Fig. 1 The BPM life-cycle shows the difference in the use of the process model

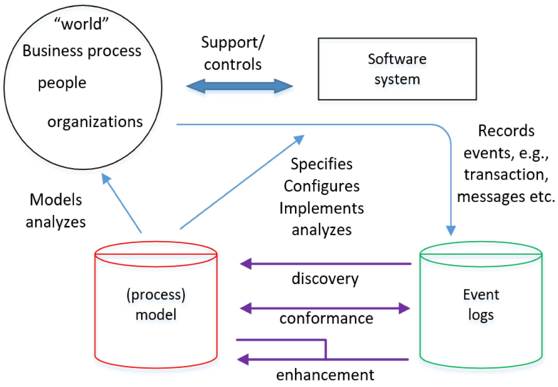

Fig. 2 The placement of the three main types of process mining

Fig. 1, Business Process Management (BPM) life-cycle, shows the difference between a process model and the life-c in a business process [7]. Fig. 2 (F. Daniel 2011) shows the three main stages in process mining: a) discovery by finding process data from the event logs; b) conformance by comparing the actual process with process rules; and c) enhancement of process model capabilities. Conformance checking is used to check whether reality is recorded in the event logs, confirmation on the model, or vice versa [4], [7], [9], [17], [18].

III. Methods

The research methodology that is shown by Fig. 3 is adopted from the L^* life-cycle proposed by van der Aalst with a few changes, a model is describing a process mining project. The method consisting of five steps:

- Step 0: Planning and justifying
  Process mining projects aim for process improvement, specifically, specific KPIs, for example, cost reductions or shorter response times;
- Step 1: Extraction
  The event log has two main conditions, namely (a) events must be ordered by time, and (b) events must be linked (that is, each event needs to refer to a specific case).
- Step 2: Creating a control flow model and relate it to the event log
  After completing step 1, the control flow model is closely linked to the event log. That is, the events in the record refer to the activities in the model. If the model fit and the logs are low (e.g., below 0.8), it is not easy to move to Stage 3. However, by definition, it should not be a problem in terms of being used for the Lasagna process.
- Step 3: Creating an integrated process model

Model capabilities are enhanced by adding perspectives to the control flow model perspective (e.g., organisation, case and time).

- Step 4: Operational support
  Step 4 of the L^* life cycle relates to three operational support activities, namely to: detect, predict, and recommend.

In this study, we use conformance checking [19], [20] to compare the SOP with the process model generated from the event logs. For data processing, we use PM4PY [21] to support a series of process mining techniques (discovery, conformance checking, and enhancement).

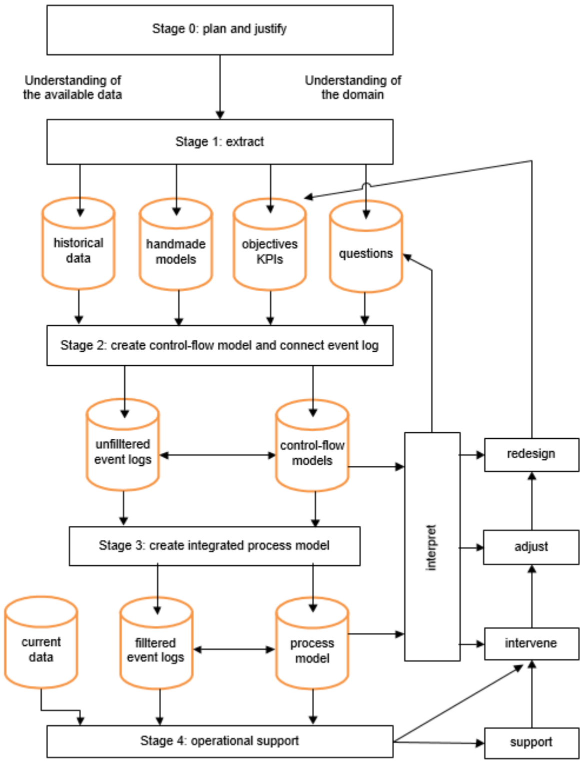
Stage 0: plan and justify
Understanding of the available data
Understanding of the domain
Stage 1: extract
historical data
handmade models
objectives KPIs
questions
Stage 2: create control-flow model and connect event log
unfilltered event logs
control-flow models
Stage 3: create integrated process model
interpret
redesign
adjust
current data
filltered event logs
process model
intervene
Stage 4: operational support
support

Fig. 3 Research method

A. Process mining α-algorithm

The Petri network in [7] is used to create a business process model within the BPM. Petri network is also included to make decision-making models. The Petri network is a bipartite graph consisting of places and transitions [7]. The network structure is static but can be governed by token tagging rules. We can route the tokens across the network. The status of the Petri network is determined by distributing tokens that pass through the place and are marked.

A Petri net is a tuple (P, T, F) where P is a definite sequence of places, T is a definite series of transitions with $P \cap T = \emptyset$, and $F \subseteq (P \times T) \cup (T \times P)$ is a circuit directed by flowing relationships. The sign in the Petri Net is a pair (N, M), where N = (P, T, F) is the Petri Net and $M \in B(P)$ is the multi-set passing P that represents the sign of the network.

- Soundness Definition (logical) When N = (P, T, F, A, l) becomes a WF-net (workflow net) with place entering *i* and place going out *o*. N is logical if and only if:
- Safeness: (N, [i]) is safe; for example, there are not many tokens at the same time in the place position.
- Proper completion: for each designation $M \in [N, [i]\rangle, o \in M$ indicates that M = [o].
- Option to complete: for each designation.
- Absence of dead parts: (N, [i]) in which there are no broken transitions (e.g., for every $t \in T$, a sequence t is possible).

Alpha Algorithm [7] is the first approach in determining the branching, namely explicit causal dependence and parallel assignment. The basic idea of the alpha algorithm (α-algorithm) is as follows. The α-algorithm input is a record of events L passing through *A*. The various activities are denoted by *A*. The output of the α-algorithm is a tag on the Petri Network α (L) = (N, M) on the WF-net, and then the initial marking can be removed and rewritten as α

(L) = N, the initial marker referred to as M = (i]. The α-algorithm will read the event logs with a specific pattern. For example, if activity a is followed by activity b, but b is never followed by a, then it is assumed that there is a cause of causal dependence between a and b. The reflection of this dependence on the Petri Network relationship, then we must place the link between a to b. Four patterns that fit in the log are defined and are known to distinguish the causes of the association. Definition of sequential relationships is based on event records, using four patterns to form a process model [7]. Thus, the business process model is constructed based on the relationship of the sequence pattern of events.

$a > L\, b$ *if and only if there is a sequence* $\sigma = \langle t_1, t_2, t_3, \ldots, t_n \rangle$ *and* $i \in \{1, \ldots, n-1\}$ *where* $\sigma \in L$ *and* $t_1 = a$ *and* $t_{i+1} = b$,
$a \longrightarrow L\, b$ *if and only if* $a > L\, b$ *and* $b \not> L\, a$,
$a \# L\, b$ *if and only if* $a \not> L\, b$ *and* $b \not> L\, a$,
$a \parallel L\, b$ *if and only if* $a > L\, b$ *and* $b > L\, a$.

B. Token-based replay

Equation (1) is to calculate the fitness on the WF-net for each trace σ by observing the tokens where p (produced tokens), c (consumed tokens), m (missing tokens), and r (remaining tokens).

$$fitness(\sigma, N) = \frac{1}{2}\left(1 - \frac{m}{c}\right) + \frac{1}{2}\left(1 - \frac{r}{p}\right) \tag{1}$$

To define the fitness on the WF-net across the event log L on the WF-net, the following (2) is used:

$$fitness(L, N) = \frac{1}{2}\left(1 - \frac{\Sigma_{\sigma \in L} L(\sigma) \times m_{N,,\sigma}}{\Sigma_{\sigma \in L} L(\sigma) \times c_{N,,\sigma}}\right) + \frac{1}{2}\left(1 - \frac{\Sigma_{\sigma \in L} L(\sigma) \times r_{N,,\sigma}}{\Sigma_{\sigma \in L} L(\sigma) \times p_{N,,\sigma}}\right) \tag{2}$$

C. Handmade Model as an initial Model Process

Process domain model, through a handmade model, describes the SOP. It is necessary to get a specific case that will be analysed. Accordingly, we can determine the initial and final activities. Fig. 4 below is the process dwelling time at the container port.

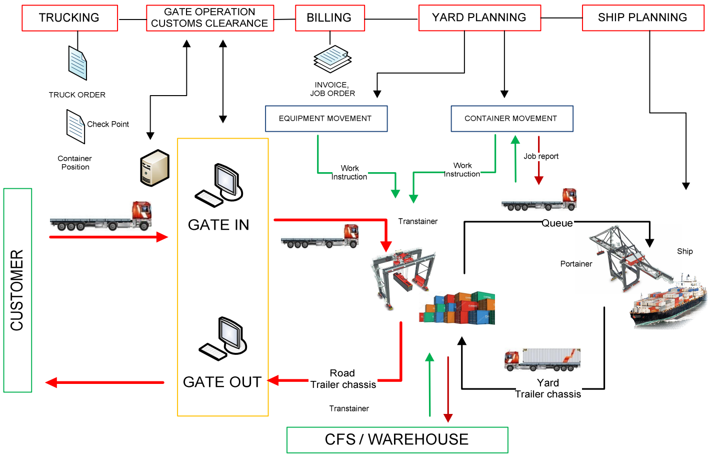

Fig. 4 Initial Handmade Model Process

IV. Results

This study [5] carried out the stages of the L^* life-cycle model (W M P der Aalst 2011) to discover (mining Lasagna processes) at a terminal container port to explore the dwelling time. From Table 1 about the dwelling time activities, there are 16 activities discovered, starting from discharge, when the container is dropped from the ship until it leaves the container port.

TABLE 1
LABELLING OF DATASETS BY ACTIVITY

Label	Activities	Label	Activities
A	Discharge	I	JComplete Striping
B	Stack Date	J	Return RS Move to Yard
C	Job BHD	K	Quarantine Job BHD
D	Move to BHD LOC	L	Move to TPFT
E	Job BHD Complete	M	Quarantine Job BHD Complete
F	Return BHD move to Yard	N	Job DEL Date
G	Job Change Status	O	Truck IN (Ext)
H	Move to CFS	P	Truck OUT

We extracted data from previous research [5], [22], then we make data selection according to the analysis's needs. The attributes used include case, label, and Time Stamp. The Algorithm 1 is below.

```
events = event_df[[‘CaseID’, ‘LABEL’, ‘Time_Stamps’]]
events.rename(columns={‘CaseID’ : ’case’, ‘LABEL’ :  ‘label’, ‘Time_Stamps’ : ‘datetime’ }, inplace=True)
```

Algorithm 1. Data Selection

Then the data filtering stage is carried out based on start activities and end activities in the import dwelling time process. The initial action in the import dwelling time process begins with activities labelled "A" and the last activities with the label "P". The filtering stage uses the following Algorithm 2.

```
from pm4py filtering import start filter
log start = get start activities (events)
start activities = apply (events, [“A”])
from pm4py filtering import end filter
end actvities = get end activities (df_start_activities)
Filtered df = apply (df_start_activities, [“P”])
```

Algorithm 2. Filtering On Start and End Activities

The next stage, Algorithm 3, is a filtering process to group the data based on variants, as shown in table 2. From the results of discovering the *L* event log, there are generated 41 traces from 102,330 cases.

```
from pm4py import case statistics
Variants count = get variant statistics (filtered_df)
Variants count = sorted (variants count, key=lambda x: x[‘case:concept:name’], reverse=True)
```

Algorithm 3. Grouping data based on variants

Then we convert the dataset into log format, for later the discovery process is carried out using the alpha miner algorithm, with the following Algorithm 4.

```
from pm4py import converter as log converter
log 1 = apply (filtered df)
```

Algorithm 4. Converting into a log format

TABLE 2
VARIANT LABEL OF TRACES IN THE LOG L

Reference	Trace	Frequency
σ_1	A,B,N,O,P	96904
σ_2	A,B,C,D,E,F,N,O,P	3734
σ_3	A,B,K,L,M,N,O,P	772
σ_4	A,B,C,D,E,N,F,O,P	238
σ_5	A,B,C,D,F,E,N,O,P	150
σ_6	A,B,C,E,N,O,P	132
σ_7	A,N,B,O,P	121
σ_8	A,B,C,D,E,N,O,P	103
σ_9	A,B,K,M,N,O,P	26
σ_{10}	A,B,N,C,D,O,P	22
σ_{11}	A,B,N,G,H,I,J,O,P	14
…	…	…
σ_{40}	A,B,C,D,E,F,C,D,F,E,N,O,P	1
σ_{41}	A,B,C,D,E,C,F,D,E,N,F,O,P	1

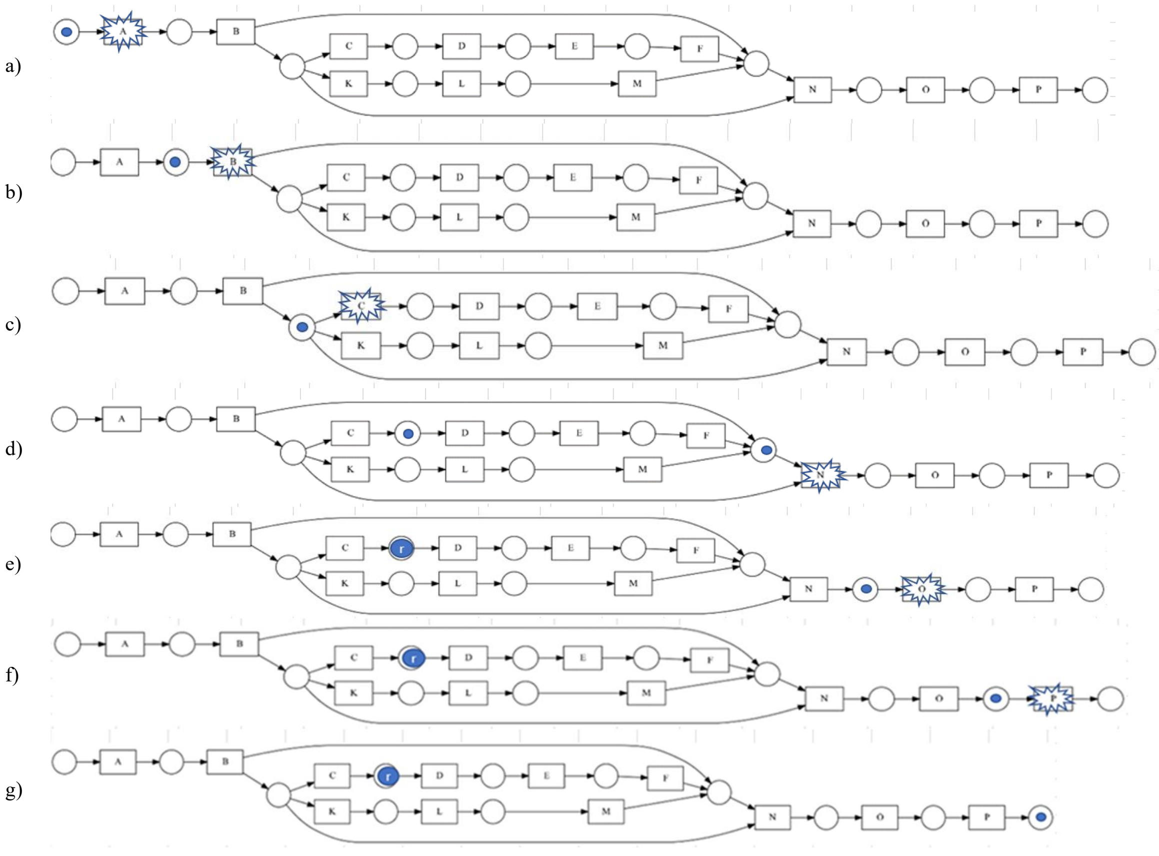

Fig. 5. fitness calculation. a) Starting trace ABCNOP. b) trace ABCNOP through A. c) trace ABCNOP through B to C. d) trace ABCNOP through C and N firing. e) trace ABCNOP through N. f) trace ABCNOP through O. g) The last replay of tokens on the trace ABCNOP.

V. DISCUSSION

In this paper, a fitness calculation is shown using the example of the one ABCNOP trace in table 2, and firing from A. In Fig. 5a, one token p=1 is produced starting from start to run activity A. The results obtained are *p=0, c=0, m=0, r=0* and *p=1, c=0, m=0, r=0.* In Fig. 5b the token passes through A to get the next sequence, which is B so that the token has been produced to p = 2 and consumed by A, which is c = 1. The results obtained are *p=2, c=1, m=0, r=0.* After B is passed in Fig. 5c. To carry out the following sequence, namely C, where the token's position is before C, three tokens have been produced, namely p = 3 and 2 tokens have been consumed, c = 2. The results obtained are *p=3, c=2, m=0, r=0.* The next step is Fig. 5d. That N then the token is positioned before because C must be passed, the token position passes C, thus increasing the token production to p = 4 and token consumption to c = 3. The results obtained are *p=4, c=3, m=0, r=0.* The next step in Fig. 5e. is O, then p = 5 is produced and token c = 4 is consumed, while the token is positioned after C doesn't move, then it becomes rest (remain), with the result r = 1. The results obtained are *p=5, c=4, m=0, r=1.* And the last activity Fig,5f., must be run is P, where the token position is before P. Thus p = 6, token consumption c = 5, and the remain token is still 1, namely r = 1. The results obtained are *p=6, c=5, m=0, r=1.* Fig.5g. The last replay of tokens in the ABCNOP process model is obtained p = 7, c = 7, and r = 1. The last results obtained are *p=7, c=6, m=0, r=1 and p=7, c=7, m=0, r=1.* Thus, conformance checking in this study compares the process model with the event log of the same process. Its purpose is to check whether the event log fits the model and vice versa.

In this research, conformance checking is done using token-based replay. The token-based reproduction will match the trace with the process model, starting from the initial activity to find which transitions were executed and which actions were missing tokens for a given process. The token-based replay will calculate the fitness value with a threshold of 1, and if the fitness value is close to 1, then the variant is close to perfect. And if the fitness value is away from 1, then the variant of the process is not comparable to excellent. Conformance checking with token-based replay using the following Algorithm 5. The results of conformance checking with token-based replay in this study can be seen in Table 3.

```
from pm4py import algorithm as token replay

replayed traces = apply (log 1, net, initial marking, final marking)

replayed traces
```

Algorithm 5. Conformance using Token-based replay

TABLE 3
CONFORMANCE CHECKING RESULTS

Variant	Produced Token (*p*)	Consumed Token (*c*)	Missing Token (*m*)	Remaining Token (*r*)	Fitness
A,B,N,O,P	8	8	0	0	1.00
A,B,C,D,E,F,N,O,P	8	8	0	0	1.00
A,B,K,L,M,N,O,P	13	12	0	1	0.96
A,B,C,D,E,N,F,O,P	8	8	0	0	1.00
A,B,C,D,F,E,N,O,P	8	8	0	0	1.00
A,B,C,E,N,O,P	8	8	0	0	1.00
A,B,K,M,N,O,P	8	8	0	0	1.00
A,B,K,M,N,O,P	11	11	2	2	0.81
A,B,C,D,E,F,C,D,E,F,N,O,P	8	8	0	0	1.00
A,B,C,D,E,F,C,D,F,E,N,O,P	8	8	0	0	1.00

The mining process results using the alpha miner Algorithm 6. will be visualised in a Petri net to display the structure contained in the dataset. The results of the Petri net can be seen in Fig. 6.

```
from pm4py import algorithm as alpha miner

net, initial marking, final marking = alpha miner apply (log 1)
```

Algorithm 6. Conforming using Token-based

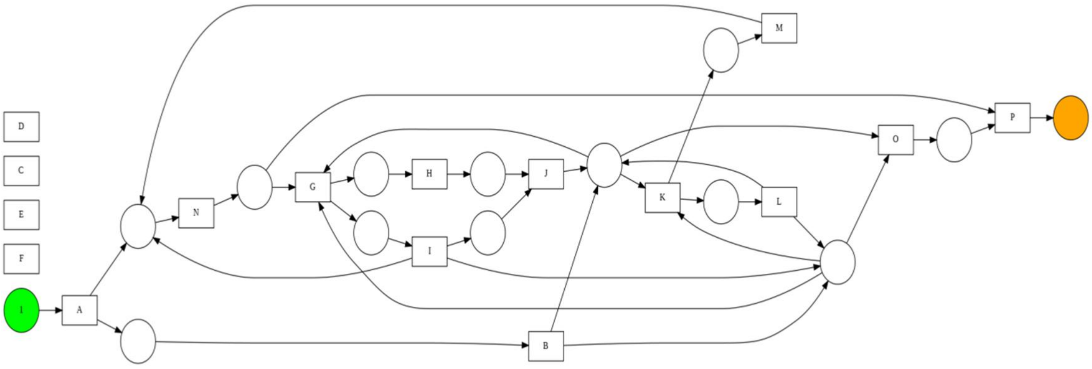

Fig. 6 Alpha Miner process model results

Fig. 6. The process model generated, activities C, D, E, F are not included in the process model structure. This happens because the existing event log data has short loops activity, including trace 33 (A, B, C, C, D, D, E, E, F, F, N, O, P).

The final evaluation stage in this research is to compare the results of the behaviour contained in the event log and the behaviour included in the process model to see the compatibility, and replay fitness is used. The goal is to calculate how the process model receives much behaviour in the event log. The evaluation of the event log and process model is shown with the following Algorithm 7. The results of this evaluation are shown in Table 4.

```
from pm4py import algorithm as replay fitness evaluator
fitness = apply (log 1, net, initial marking, final marking,
variant= variants TOKEN BASED)
```

Algorithm 7. Evaluation of the fitness

TABLE 4
EVENT LOG EVALUATION RESULTS AND PROCESS MODEL

Average Trace Fitness	0.999610282059692
Log Fitness	0.9993923001223441
Percentage Of Fitting Traces	99.16349066744846

VI. CONCLUSIONS

Based on the *L** life-cycle methodology used in this study, the process mining using Alpha miner is carried out to produce a process model. The process model is a control flow of activities running on the container port. The result of the fitness calculation is more than 0.8, which is 1.0, which indicates that the resulting control flow is good.

From the results of the fitness calculation, it is found that the process model and event logs match, which means that the process model is following the reality of transactions recorded in the event log. Thus, the event log corresponds to the resulting process model. This study focuses on process discovery techniques using the Alpha miner algorithm. So, in the following research development, process discovery can be made using Alpha plus, which can overcome short loops.

Author Contributions: *Bambang Jokonowo*: Conceptualization, Methodology, Writing - Original Draft, Writing - Review & Editing, Supervision, Software, Investigation, Data Curation, *Nenden Siti Fatonah*: Conceptualization, Methodology, Writing - Original Draft, Software, Data Curation, *Emelia Akashah Patah Akhir*: Conceptualization, Writing - Review & Editing, Supervision.

Funding: This work was supported by Mercu Buana University under [INTERNATIONAL JOINT RESEARCH Mercu Buana University and Universiti Teknologi PETRONAS (UTP)] Grand number: 02-5/754/B-SPK/II/2021.

Acknowledgments: The acknowledgements are given to Amin Jamal as IT Manager, Port Operation Team, and Terminal Petikemas Surabaya (TPS) Board of Directors, and Rivalno as a member of Research Group Business Process Intelligence UMB.

Conflicts of Interest: The authors declare no conflict of interest.

REFERENCES

[1] W. M. P. der Aalst, *Process Mining: Discovery, Conformance and Enhancement of Business Processes*, 1st ed. Springer Publishing Company, Incorporated ©2011, 2011.

[2] Y. A. Effendi and N. Nuzulita, "Process Discovery of Business Processes Using Temporal Causal Relation," *Journal of Information Systems Engineering and Business Intelligence*, vol. 5, no. 2, p. 183, Oct. 2019, doi: 10.20473/jisebi.5.2.183-194.

[3] H. van der Aa, H. Leopold, and H. A. Reijers, "Efficient process conformance checking on the basis of uncertain event-to-activity mappings," *IEEE Transactions on Knowledge and Data Engineering*, vol. 32, no. 5, pp. 927–940, 2020, doi: 10.1109/TKDE.2019.2897557.

[4] S. J. van Zelst, A. Bolt, M. Hassani, B. F. van Dongen, and W. M. P. van der Aalst, "Online conformance checking: relating event streams to process models using prefix-alignments," *International Journal of Data Science and Analytics*, vol. 8, no. 3, pp. 269–284, 2019, doi: 10.1007/s41060-017-0078-6.

[5] B. Jokonowo, R. Sarno, S. Rochimah, and B. Priambodo, "Process Mining : Measuring Key Performance Indicator Container Dwell Time," *Indonesian Journal of Electrical Engineering and Computer Science*, no. x, pp. 31–41, 2019, [Online]. Available: http://ijeecs.iaescore.com

[6] B. Jokonowo, J. Claes, R. Sarno, and S. Rochimah, "Process Mining in Supply Chains: A Systematic Literature Review," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 8, no. 6, 2018, doi: 10.11591/IJECE.V8I6.PP%P.

[7] W. van der Aalst, "Data Science in Action," in *Process Mining*, Berlin, Heidelberg: Springer Berlin Heidelberg, 2016, pp. 3–23. doi: 10.1007/978-3-662-49851-4_1.

[8] H. van der Aa *et al.*, "Monotone conformance checking for partially matching designed and observed processes," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8, no. December, pp. 81–88, 2020, doi: 10.1109/ICPM.2019.00022.

[9] W. M. P. van der Aalst and H. M. W. Verbeek, "Process discovery and conformance checking using passages," *Fundamenta Informaticae*, vol. 131, no. 1, pp. 103–138, 2014, doi: 10.3233/FI-2014-1006.

[10] E. Asare, L. Wang, and X. Fang, "Conformance Checking: Workflow of Hospitals and Workflow of Open-Source EMRs," *IEEE Access*, vol. 8, pp. 139546–139566, 2020, doi: 10.1109/ACCESS.2020.3012147.

[11] E. Skoufias, M. Rabassa, S. Olivieri, and M. Brahmbhatt, "Why Cargo Dwell Time Matters in Trade," *Management*, vol. 2010, no. September, pp. 1–5, 2012, doi: 10.2139/ssrn.1686004.

[12] W. van der Aalst *et al.*, "Process Mining Manifesto," pp. 169–194, 2012, doi: 10.1007/978-3-642-28108-2_19.

[13] M. de Leoni and A. Marrella, "How Planning Techniques Can Help Process Mining: The Conformance-Checking Case Multi-perspective Process Mining View project How Planning Techniques Can Help Process Mining: The Conformance-Checking Case," 2017. [Online]. Available: https://www.researchgate.net/publication/322917255

[14] W. Grossmann and S. Rinderle-Ma, *Fundamentals of Business Intelligence, Data-Centric Systems and Applications*. © Springer-Verlag Berlin Heidelberg, 2015. doi: DOI 10.1007/978-3-662-46531-8_1.

[15] W. van der Aalst, "Using process mining to bridge the gap between BI and BPM," *Computer*, vol. 44, no. 12, pp. 77–80, 2011, doi: 10.1109/MC.2011.384.

[16] A. Rozinat and W. M. P. van der Aalst, "Conformance checking of processes based on monitoring real behavior," *Information Systems*, vol. 33, no. 1, pp. 64–95, 2008, doi: 10.1016/j.is.2007.07.001.

[17] V. Naderifar, S. Sahran, and Z. Shukur, "A review on conformance checking technique for the evaluation of process mining algorithms," *TEM Journal*, vol. 8, no. 4, pp. 1232–1241, 2019, doi: 10.18421/TEM84-18.

[18] S. Dunzer, M. Stierle, M. Matzner, and S. Baier, "Conformance checking: A state-of-the-art literature review," 2020, doi: 10.1145/3329007.3329014.

[19] J. Carmona, B. van Dongen, A. Solti, and M. Weidlich, *Conformance checking: Relating processes and models, relating processes and models*. 2018. doi: 10.1007/978-3-319-99414-7.

[20] M. F. Sani, M. Kabierski, S. J. van Zelst, and W. M. P. van der Aalst, "Model Independent Error Bound Estimation for Conformance Checking Approximation," pp. 1–16, 2021, [Online]. Available: http://arxiv.org/abs/2103.13315

[21] A. Berti, S. J. van Zelst, and W. van der Aalst, "Process Mining for Python (PM4Py): Bridging the Gap Between Process- and Data Science," May 2019, [Online]. Available: http://arxiv.org/abs/1905.06169

[22] B. Jokonowo, R. Sarno, and S. Rochimah, "Extracting Audit Trail Data of Port Container Terminal for Process Mining," 2019.