An Approach for Mining Multiple types of Silent Transitions in Business Process

Li-li Wang¹, Xian-wen Fang¹, Chi-feng Shao², Esther Asare¹
¹College of Mathematics and Big Data, Anhui University of Science and Technology, Huainan 232000 China
²College of Information & Network Engineering, Anhui Science and Technology University, Bengbu 233100 China

Corresponding author: Xian-wen Fang (e-mail: xwfang@aust.edu.cn).

This work was supported in part by the National Natural Science Foundation, China (No. 61572035, 61402011), the Leading Backbone Talent Project in Anhui Province, China (2020-1-12), the Natural Science Foundation of Anhui Province, China (No. 2008085QD178), Anhui Province Academic and Technical Leader Foundation (No. 2019RZ39), Anhui Province College Excellent Young Talents Fund Project of China (gxyqZD2020020), and the Open Project of the Key Laboratory of Embedded System and Service Computing Ministry of Education (No. ESSCKF2018-04).

ABSTRACT The purpose of process discovery is to construct a process model based on business process execution data recorded in an event log. Many situations may lead to silent transitions that appeared in the process model, while the execution of silent transitions is not recorded in event logs. Therefore, mining silent transitions has been one of the difficult problems in process mining. Existing approaches have some limitations on discovering the silent transition in concurrent structures and may produce many redundant silent transitions which make discovered process model complicated. A novel approach to discover multiple types of silent transitions from an event log is presented in the paper. The basic behavior relationship between activity pairs based on the event log is used to construct the process model with silent transitions of and-gateway type and loop type. Meanwhile, the technique of behavior distance is proposed to discover silent transitions of skip type. Finally, the process model with multiple types of silent transitions is obtained. Experimental results show that the proposed approach can find multiple types of silent transitions correctly, and the number of redundant silent transitions is much less than the existing methods. Meanwhile, it significantly improves the F-measure of the model.

INDEX TERMS behavior distance, labeled Petri net, process discovery, silent transition.

I. INTRODUCTION

Nowadays, information system records the real execution of the business process in the form of event logs. The goal of process mining is to discover, monitor and enhance real life processes by automatically extracting valuable information from these event logs. It includes three main domains, i.e., process discovery, conformance checking and process enhancement. However, process discovery is the necessary technique for the other domains, which automatically constructs process models by applying the different discovery techniques on event logs [1]. Until recently, dozens of process discovery algorithms have been proposed [2-4]. Whereas, many problems need to be solved in process mining, such as short loops, duplicated transitions, invisible (or silent) transitions, non-free-choice constructs, noises, infrequent behaviors, incompleteness, etc [5]. In addition, privacy-preservation issues of cross-organization business process mining [6,7], efficiency problem when dealing with large-scale event logs [8], process model discovery and process similarity measure considering both control-flow structures and data-flow information [9]. However, in this paper, we focus on the detection of silent transitions from event logs. Various factors in real life may cause silent transitions to appear in the process model, and these silent transitions only play a role of routing but cannot be ignored. Silent transitions are difficult to discover because they do not appear in any event trace. The main reasons for leading to silent transitions are as follows [10]:

• There are some actual tasks that can be allowed to be missing in some event traces.
• There are meaningless tasks only used for routing purposes in process models.
• The enactment services of process models allow skipping or redoing the current task and jumping to any previous task. But such execution logic is not expressed in the control logic of the process model.

Until now, there are several mining approaches that are capable of discovering silent transitions [10-19], such as α#
algorithm[10,11], genetic algorithm[12], α$[15]$. Inductive Mining (IM) algorithm [2], Coupled Silent Markov Model-Nonfree Choice Invisible Task(CHMM-NCIT)[17], etc. These mining algorithms use techniques such as directly-follows relationships, Markov model, or genetic algorithm to detect silent tasks. However, some yield a lot of redundant silent tasks, and some fail to detect silent tasks that should be appeared in the concurrent structure. Thus, the existing process discovery algorithms have difficulties to discover possible silent tasks from the given event logs accurately and guarantee to return the appropriate result. They pay less attention to the behavioral relationship between activities in the event logs when detecting silent transitions. Thus, a novel approach to discover silent transitions based on the behavior distance is presented in the paper. The proposed approach can discover multiple types of silent transitions and deal with silent transitions that appeared in concurrent structures. Meanwhile, it will not produce a large number of redundant silent transitions. Results of experiments on event logs show that the proposed method is promising compared to existing methods. The main works of the paper are as follows:

1. We present a new technique of behavior distance which can quantify the behavior relationship between activities in the event logs. Subsequently, the behavior distance between activities based on log, model, and concurrent structure is presented respectively to discover silent transitions accurately.

2. The traditional behavioral profile is refined to more accurately capture the behavior relationship between activities. We construct the process model with silent transitions by analyzing the behavior relationship between activities in the event logs.

3. The novel approach to discover multiple silent transitions is proposed based on behavior distances and basic behavioral relations of event logs. The proposed approach can find multiple types of silent transitions correctly, and the number of redundant silent transitions is much less than the existing methods.

The remainder of the paper is structured as follows. Section 2 reviews works on the silent task. Section 3 introduces a motivation example. Section 4 presents the preliminary concepts. The approach for discovering silent transitions of and-gateway type and loop type is given in Section 5. Further on, the technology of discovering silent transitions of skip type is described in Section 6. Section 7 shows the evaluation results of the proposed approach. Finally, Section 8 concludes the paper and discusses future work.

II. RELATED WORKS

Over the last decade, there are several mining techniques which support the detection of silent transitions, such as IM algorithm [2], α$ algorithm [10,11], genetic algorithm[12], α$[15], Coupled Silent Markov Model- Invisible Task(CHMM-IT)[16], Coupled Silent Markov Model- Nonfree Choice Invisible Task(CHMM-NCIT)[17], etc. The IM algorithm [2] obtains a process model with silent transitions using the technique of process tree cutting. Still, it may yield too many redundant silent transitions, which makes the process model very complicated. The literature [11] gives a mining method of a process model with prime invisible tasks based on the literature [10]. This approach detects invisible tasks by identifying the mendacious dependency, which is separated from the causal dependency. Therefore, if the tasks have no directly-follows relation in the event log, then the silent task between them will not be detected. Thus, it is difficult to detect the silent transition appeared in concurrent structure. In [12], the authors propose the genetic mining algorithm, which supports the detection of invisible tasks, duplicated tasks and non-free choice constructs. Whereas this algorithm needs many user-defined parameters and it cannot always guarantee to return the most appropriate results. A Synchronization-based mining algorithm, which is a synchronization-based model of workflow logic and workflow semantics is proposed in [13]. The authors state that short-loops and invisible tasks can be dealt with at ease. However, neither the original model nor the mined model contains any invisible task. Literature [14] presents a decisions mining approach to mine decisions from process logs, which emphasizes detecting data dependencies that affect the routings of cases. When interpreting the control-flow semantics at a decision point, the authors propose an approach to identify decision branches starting from invisible tasks. However, not all kinds of silent tasks can be dealt with by this approach. The literature [15] provides the α$ algorithm to handle silent transitions in non-free-choice Petri net, but it still cannot detect the silent transition that appeared in concurrent structure. Literatures [16,17] present a method that utilizes silent Markov to construct process models with a free and non-free choice construct containing invisible prime tasks from incomplete event log, both Coupled Silent Markov Model-Nonfree Choice Invisible Task (CHMM-NCIT) and Coupled Silent Markov Model-Invisible Task (CHMM-IT) use Coupled Silent Markov Model (CHMM) and use Baum-Welch algorithm to determine the weights of variables of CHMM. However, the time complexity of these algorithms is very high. In [18], the authors propose a method to construct silent task and silent task in non-free choice relation when converting a declarative model into an imperative model in the form of Petri net model. Therefore, this method cannot discover silent task when constructing the process model from the event logs. In literature [19], a graph-based algorithm is proposed to discover silent transitions. However, when event logs are very large, it is difficult to get database graphs to express the dependency between events.

All methods mentioned above, none of them concentrate on the behavioral relation between activities when detecting the silent task, and they still cannot handle all kinds of silent transitions well without too many redundant silent transitions. This paper presents a novel approach to discover silent transitions based on behavioral
relationship without generating a large number of redundant silent transitions.

III. MOTIVATION

In the following, we will illustrate the limitation of existing approach to mine silent transitions with an example. For the event log \( L = \{\sigma_1, \sigma_2, \sigma_3, \sigma_4, \sigma_5, \sigma_6, \sigma_7, \sigma_8, \sigma_9, \sigma_{10}, \sigma_{11}, \sigma_{12}, \sigma_{13}\} \)
where, \( \sigma_1 = \text{ACDIJKM} \), \( \sigma_2 = \text{ACDIJKKJM} \), \( \sigma_3 = \text{ADILJKM} \), \( \sigma_4 = \text{BDLJIKKM} \), \( \sigma_5 = \text{BCDLMJUM} \), \( \sigma_6 = \text{ADJIJKJM} \), \( \sigma_7 = \text{ADLJKJYJKKJM} \), \( \sigma_8 = \text{AEFHJKMK} \), \( \sigma_9 = \text{AEGLHJKM} \), \( \sigma_{10} = \text{BGEGHJM} \), \( \sigma_{11} = \text{BEGFHJKM} \), \( \sigma_{12} = \text{BEGFGFFGJHJM} \), \( \sigma_{13} = \text{AEHGFGFFHJKM} \). The frequency of traces is not considered in the paper. The process models shown in Figure 1 and Figure 2 are obtained by applying the approach proposed in the literature [3] and literature [12] on the above log respectively. Applying the proposed method in this paper, the process model can be yielded shown in Figure 3.

![Figure 1](image1.png)
**FIGURE 1.** A process model \( M \) obtained by a\# algorithm.

![Figure 2](image2.png)
**FIGURE 2.** A process model \( M' \) obtained by IM algorithm.

![Figure 3](image3.png)
**FIGURE 3.** A process model \( M^* \) obtained by the proposed approach.

The literature [3] has difficulties to discover the silent transition between activities pairs that have no directly-follows relation, such as the silent transition \( t_i \) shown in Figure 3. The Inductive Mining method [12] produces lots of redundant silent transitions, which lead to the model complex, incomprehensible and inaccurate analysis results. To address the aforementioned problems, a novel method of discovering a process model with non-redundant silent transitions is presented in this paper, which overcomes the drawbacks of existing approaches to mining silent transitions. Meanwhile, the effectiveness of the method through a set of experiments is validated.

IV. PRELIMINARIES

The basic definitions and related knowledge of Petri net will not be introduced in this paper. More details can be found in the literature [20]. The basic knowledge of process mining can be found in the literature [1]. Here, we only present basic preliminaries and notations, used throughout the paper.

**Definition 1** *(Labeled Petri Net)* A labeled Petri net is a 4-tuple \( N := (P, T, F, \lambda) \), where \( (P, T, F) \) is a Petri net. \( \lambda \) represents the universe of labels to describe actions that have explicit meaning. \( \tau \) represents the particular label that has no domain interpretation, where \( \tau \notin \lambda : T \rightarrow \lambda \) \((\lambda \cup \{\tau\})\) is a function that assigns labels to transitions. If \( \lambda(t) \neq \tau \), then \( t \) is observable. Otherwise, \( t \) is silent or invisible.

A labeled net system \( S := (P, T; F, \lambda, M_0) \) is a labeled net \( N := (P, T; F, \lambda) \) with initial marking \( M_0 \). The semantics of a labeled net is the same as for an unlabeled net. For each occurrence sequence \( \sigma = t_1t_2 \cdots t_n \) of \( S \) induces the labeled occurrence sequence \( l(\sigma) = l(t_1)l(t_2) \cdots l(t_n) \), obtained by replacing each transition \( t_i \) by its label \( l(t_i) \) and omitting all \( \tau \) from the result by projection onto \( \lambda \). We say that \( N \) can replay a \( \log \) \( L \) iff each \( \sigma \in L \) is a labeled occurrence sequence of \( N \).

**Definition 2** *(Labeled Execution Sequence or Labeled Occurrence Sequence)* Let \( S := (N, M_0) \) be a labeled net system, where \( N := (P, T, F, \lambda) \). All labeled execution sequences of a net system \( S \) are all lists of the form \( \{a_i\}^{(\lambda \setminus \tau)} \{a_f\} \) that follow the executable semantic of \( S \), where \( a_i,a_f \) is an artificial start transition and end transition respectively. All labeled execution sequences of \( S \) are denoted as \( T(S) \).

**Definition 3** *(Directly-follows Relation (log))* Let \( L \) be an event log over \( T \). For \( a, b \in T \) , if existing a trace \( \sigma \in L \) and \( \sigma = t_1t_2 \cdots t_n \) such that \( t_i = a, t_{i+1} = b \) (where \( i \in \{1,2,\ldots,n-1\} \) ), we say that there is a directly-follows relation between \( a \) and \( b \), which is written as \( a \rightarrow_L b \).

**Definition 4** *(Weak Order(Log))* Let \( L \) be an event log over \( T \). For \( a, b \in T \) , if existing a trace \( \sigma \in L \) and \( \sigma = t_1t_2 \cdots t_n \) such that \( t_i = a, t_j = b \) (where \( 1 \leq i < j \leq n \) ), we say that there is a weak order relationship between \( a \) and \( b \), which is written as \( a \rightarrow_L b \).

**Definition 5** *(Free-choice)* A Petri net is free-choice if any two transitions sharing an input place have identical input sets, i.e., \( 'T_i \cap 'T_j = \emptyset \) implies \( 'T_i \subseteq 'T_j \) for any \( t_i, t_j \in T \). A Petri net is sound, if it is safeness, proper completion and has no dead transitions, meanwhile, all transitions are on the path from input place \( i \) and output place \( o \).

V. CONSTRUCTING A PROCESS MODEL WITH SILENT TRANSITIONS OF AND-GATEWAY AND LOOP TYPE

On the basis of literature [23], this section further points out how to associate the behavioral characteristic relationship
between activities in the event log and their structural relationship in the process model through the behavioral relationship and behavioral distance in the log. The correctness of the corresponding relation is proved by propositions, which provides a theoretical basis for finding the initial process model with silence transition of and-gateway type and loop type from event log. Moreover, the approach to discover silent transitions of and-gateway and loop type from event log is provided.

Obviously, directly-follows relation and weak order relation between activity pairs are qualitative analysis methods to behavior relation. However, behavioral distance is presented in the following, which realizes the quantitative analysis of behavioral relations.

**Definition 6 (Trace-based Behavior Distance)** Let $L$ be an event log over $T$. For $a, b \in T$, the behavior distance between $a$ and $b$ is defined as:

$$BDis(a, b, \sigma) = \begin{cases} \{k | 1 \leq i \leq |\sigma| \land (i \land \sigma(i) = a \land \sigma(i + k) = b) \} & \text{if } a \geq_{L} b \\ \{ -BDis(b, a, \sigma) \} & \text{if } b \geq_{L} a \land a \not\geq_{L} b \end{cases}$$

(1)

**Definition 7 (Log-based Maximum Behavior Distance and Minimum Behavior Distance)** Let $L$ be an event log over $T$. For $a, b \in T$, log-based minimum behavior distance $BD_{min}(a, b, L)$ and log-based maximum behavior distance $BD_{max}(a, b, L)$ are defined as:

$$BD_{min}(a, b, L) = k \iff 3\sigma \in L, BDis(a, b, \sigma) < k,$$

(2)

$$BD_{max}(a, b, L) = k' \iff 3\sigma \in L, BDis(a, b, \sigma) > k'. $$

(3)

If an activity $a$ and an activity $b$ are in an exclusive relationship, i.e., they can never both appear in a trace, then $BD_{min}(a, b, L) = BDis_{max}(a, b, L) = \infty$. If $a \geq_{L} b$ and $b \not\geq_{L} a$ holds in any trace, then $BD_{min}(b, a, L) = -BDis_{max}(a, b, L)$ and $BD_{max}(b, a, L) = -BDis_{max}(a, b, L)$.

Given an event log $L$, $A_{t}$ represents a set of activities appeared in the log $L$. Let $|A_{t}| = n$, $BDMatrix$ is a $n \times n$ dimensional matrix which denotes the behavioral distance between all activities based on the event log. For all $i, j \in \{1, \ldots, n\}$, the $i$-th row corresponds to activity $a$ and the $j$-th column corresponds to activity $b$, the value of the $i$-th row and $j$-th column of the matrix denoted as $M_{ij} = x_{i, j}$, where $x_{i, j} = BD_{min}(a, b, L)$, $y_{i, j} = BD_{max}(a, b, L)$.

The behavior distance matrix of the event log $L_{l}$ shown in Section 2 is in Figure 4.

**Definition 8 (Basic Behavior Relation based on Log)**

Let $L$ be an event log over $T$, let $a, b \in T$.

- The causal relation $\rightarrow_{L}$, iff $a \geq_{L} b, b \not\geq_{L} a$ and $BD_{min}(a, b, L) = BD_{min}(a, b, L) = 1$.
- The strict order relation $\rightarrow_{L}$, iff $a \geq_{L} b, b \not\geq_{L} a$ and $BD_{max}(a, b, L) > 1$.
- The exclusiveness relation $\perp_{L}$, iff $a \not\geq_{L} b$ and $b \not\geq_{L} a$.
- The concurrent interleaving order relation $\parallel_{L}$, iff $a \geq_{L} b, b \geq_{L} a, b \not\geq_{L} a$.
- The loop interleaving order relation $\|_{L}$, iff $a \geq_{L} b, b \geq_{L} a, b \geq_{L} a$.

The set $\{\rightarrow_{L}, \rightarrow_{L}, \oplus_{L}, \parallel_{L}, \|_{L}\}$ is the basic behavior relation of $L$.

For the event log $L_{l}$ shown in Section 2, the basic behavior relationship between activities is shown in Figure 5.

**Definition 9 (Silent Transition type)**

Let $N = (P, T, F, \lambda)$ be a labeled Petri net, $A \cup \tau$ represent all labels of activities. For $t_{1}, t_{2} \in A \cup \tau$, if

1. $t_{1} \rightarrow_{N} t_{2}$, $t_{1}^{*} \cap T \neq \emptyset$ and $t_{1}^{*} \cap t_{2}^{*} \neq \emptyset$.

2. $t_{1} \rightarrow_{N} t_{2}$, $t_{1}^{*} \cap T \neq \emptyset$ and $t_{1}^{*} \cap t_{2}^{*} \neq \emptyset$.

A literature [11] indicates that the mendacious dependencies are important for the discovery of silent transitions. However, classical behavioral profiles do not distinguish mendacious dependency from directly-follows dependency. As we know that the cyclic structures have a substantial impact on the interleaving behavior relations, for example, two exclusive transitions inside a cycle. Therefore, it cannot determine the structural characteristics between activities, such as true concurrency and cyclic structure, when activity pairs are in interleaving behavior relations. Obviously, existing behavioral profile is too rough. It is very important to identify causal dependency, mendacious dependency and different interleaving order relationships for the detection of silent transition. Therefore, Definition 8 presents basic behavior relation based on log.

**Figure 4. The behavior distance matrix of $L_{l}$**

**Figure 5. Basic behavior relationship between activities based on $L_{l}$**

Definition 9 presents the definition of various types of silent transition according their structural character.
Proposition 1 If activity $a$ and activity $b$ are in causal dependence based on log, then they are in structural concurrency. 

Proof. We prove it in three cases. Case 1: Assume $a \not= b$ and $|a| \cap |b| = 1$. Obviously, this leads to a contradiction with $a \rightarrow b$. Case 2: Assume $a = b$ and $|a| \cap |b| \not= 1$. This indicates that there are multiple connection places between activity $a$ and activity $b$. Only preserving one place by deleting others places does not affect the behavioral relationships between activity $a$ and activity $b$. Case 3: Assume $a \not= b$ and $|a| \cap |b| = \emptyset$. Since $a \rightarrow b$, we infer $a^* \cap b \not= \emptyset$. Without loss of generality, we just discuss the simplest case. Let $a^* \cap b = p$, we assume $|a^*| = 2, |b| = 1$, let $a^* = \{p,p\}$, $\hat{p}^* = c$. When the activity $a$ is fired, activity $b$ and activity $c$ are in concurrent. Hence, there exists an execution sequence in the form of $\cdots abc\cdots$, which produces $BD_{\max}(a,b) > 1$. Therefore, this leads to a contradiction.

Proposition 2 If activity $a$ and activity $b$ are in strict order relation based on log, then there exists a path composed of transitions and places between activity $a$ and activity $b$ in a Petri net.

Proof. According to the definition of the strict order, it is obvious that it is true.

Proposition 3 For any sound free-choice system it holds that, if activity $a$ and activity $b$ are in exclusiveness relation based on log, then exclusiveness and structural exclusiveness coincide.

Proof. Assume activity $a$ and activity $b$ are not in structural exclusiveness. As $a \rightarrow b$, it is impossible that activity $a$ and activity $b$ are in structural concurrent or structural cycle. If a path from activity $a$ to activity $b$ exists, from $a \rightarrow b$, we know that the possible structure between them is shown as Figure 7. i.e. $\exists c$, satisfy $a \cap c \not= \emptyset$, $c \cap b \not= \emptyset$. Clearly, activity $b$ is not in choice-free structure, which leads to a contradiction.

Proposition 4 If activity $a$ and activity $b$ are in the concurrent interleaving order relation based on log, then they are in structural concurrent.

Proof. We assume activity $a$ and activity $b$ are not in concurrent structure in process model. From the definition of concurrent interleaving order relation, we know that activity $a$ and activity $b$ are either in exclusive structure or in sequential structure, so we have to consider two cases. For the former, without losing generality, we only consider the concurrent relation of length 1. In order to guarantee $a \succeq_L b, b \succeq_L a$ both hold and the system is sound, the possible structures are as shown in Figure 8. It is obviously that $b \succeq_L a, a \succeq_L b$ also hold except $a \succeq_L b, b \succeq_L a$. That yields a contradiction with $a \parallel b$. For the latter, as they are in sequential structure, then the possible structures are as shown in Figure 9. It is obviously that $b \succeq_L a, a \succeq_L b$ both hold. That yields a contradiction with $a \parallel b$. As both assumptions lead to contradictions, we know they are structural concurrent.
complex relations can be transformed into basic structural relations through clustering.

1. Case 1: As \( \exists t_1, t_1 \rightarrow a, t_1 \rightarrow b \) hold, meanwhile, activity \( a \) and activity \( b \) are exclusive inside cyclic structure. The possible structures are shown in Figures 10(a), 10(b), 10(c). In Figure 10(a), \( \text{BDist}(t_1, b, L_{min}) \geq \text{BDist}(t_1, a, L_{min}) + 1 \) doesn’t hold when the occurrence sequence in the form of \( ...ta...\), which leads to a contradiction. In Figure 10(b), \( \forall \sigma \in L, |\text{OccurTime}(a, \sigma_r) - \text{OccurTime}(b, \sigma_r)| \leq 1 \) doesn’t hold when the occurrence sequence in the form of \( ...taaaa... \), which leads to a contradiction. In Figure 10(c), \( \text{BDist}(t_1, b, L_{min}) \geq \text{BDist}(t_1, a, L_{min}) + 1 \) holds. When activity \( a \) and activity \( b \) are both in loop-body or loop-redo, \( \forall \sigma \in L, \text{OccurTime}(a, \sigma_r) = \text{OccurTime}(b, \sigma_r) \) holds. When activity \( a \) is in loop-body and activity \( b \) is in loop-redo, \( \forall \sigma \in L, \text{OccurTime}(a, \sigma_r) = \text{OccurTime}(b, \sigma_r) + 1 \) holds. Therefore
\[ \forall \sigma \in L, |\text{OccurTime}(a, \sigma_r) - \text{OccurTime}(b, \sigma_r)| \leq 1 \]
always holds in the Figure 10(c). Hence, for case 1, activity \( a \) and activity \( b \) must be sequential relation inside a cyclic structure.

2. Case 2: As \( \exists t_1, t_1 \rightarrow a, t_1 \rightarrow b \) hold. Moreover, for \( \forall \sigma \in L \), there is no dependencies between \( \text{OccurTime}(a, \sigma_r) \) and \( \text{OccurTime}(b, \sigma_r) \), i.e., the occurrence frequency between \( a \) and \( b \) is independence, then activity \( a \) and activity \( b \) are exclusive inside cyclic structure.

3. For case 3, as \( \exists t_1, t_1 \rightarrow a, t_1 \rightarrow b \) hold. Moreover, for \( \forall \sigma \in L \), \( \text{OccurTime}(a, \sigma_r) = \text{OccurTime}(b, \sigma_r) \), then activity \( a \) and activity \( b \) are concurrent inside cyclic structure.

Proof. The above three cases are respectively proved. Without loss of generality, only the basic structural relations among activities are considered here, and other
FIGURE 10. The possible structures between activity $a$ and activity $b$ when they are in the loop interleaving order relation.

**Lemma 1** If activities are in concurrent interleaving order relation, then there are silent transitions of and-split and and-join type.

**Proof.** Obviously it is correct.

Although the silent transition of and-gateway type added by Lemma 1 will not affect the behavior relation of any other activities of the net system, it may produce many redundant silent transitions of and-gateway type. Algorithm 1 will remove these redundant silent transitions of the and-gateway type according to structural features between the silent transition and its pre- or post-transitions.

**Theorem 1** Let $L$ be the event log over $T$. For $T' \subseteq T$, if $T'$ is a maximal subset of activities that is in the loop interleaving order relation and $\forall t \in T'$ belongs to the part of loop-body, then there exists a silent transition of loop type that belongs to the part of loop-redo.

**Proof.** We assumed that there is no silent transition of loop type, i.e., there exists a visible activity $t'$ as a loop-redo transition. As $\forall t \in T'$ belongs to the part of the loop-body, the possible net construction of $T'$ and $t'$ is shown in Figure 11. It is easy to know that $\forall t \in T'$ and $t'$ are in the loop interleaving order relation. Hence, the maximal subset of activity that is in the loop interleaving order relation is $T' \cup t'$, which contradict with the condition. Therefore, Theorem 1 is proven to be correct.
Meanwhile, silent transitions of and-gateway and loop type can be discovered using Lemma 1 and Theorem 1. Finally, the redundant silent transitions of and-gateway types are removed from the process model.

SubSet and subModules are activity subset getting from a behavior relation Matrix and sub-module set composed of these activity subsets respectively. Function Construct(S, T, structural exclusiveness) construct a sub-module with S and T in structural exclusiveness, and others are similar. InsertSilent(and-type) discovers a silent transition of and-type, and others are similar. Merge(Md₁, Md₂, Md₃) combine sub-modules Md₁ and Md₂ to get a larger sub-module Md₃. Step1-Step2 creates a behavior distance Matrix and a behavior relation Matrix based on event logs. Step4-Step30 constructs several sub-modules with silent transitions consisting of activity subsets. In Step31-Step36, the sub-modules are combined into a complete process model through causal dependency and strict relation in the behavior relation matrix. Step37-45 removes the redundant silent transitions of and-gateway type.

The execution of Algorithm 1 is illustrated using the event log \( L_1 \) in Section 2. The behavior distance matrix \( BDisMatrix_{a\sb} \) and basic behavior relationship matrix \( BRelMatrix_{a\sb} \) are shown in Figure 4 and Figure 5, respectively, in Section 2. The four sub-modules shown in Figure 12 are constructed according to Step4-30. Subsequently, they are merged into a whole process model by using Step31-36, as shown in Figure 13.

VI. MINING SILENT TRANSITIONS OF SKIP TYPE

Section 4 has shown how to construct a process model with silent transitions of and-gateway type and loop type. This section discusses how to discover silent transitions of skip type based on the initial process model constructed by Algorithm 1, thereby further optimize the initial process model.

Definition 10 (Model-based minimum behavior distance)

Let \( S = (N, M_\theta) \) be a net system, where \( N = (P, T, F, \lambda) \) is a labeled Petri net, \( \mathcal{A} \cup \tau \) is all transition labels. For \( \forall a, b \in \mathcal{A} \), the model-based minimum behavior distance between \( a \) and \( b \) is defined as

\[
BD_{\text{min}}(a, b, S) = \{ k \mid \sigma \in \mathcal{T}(S) \mid BDis(a, b, \sigma) \geq k \}.
\]

Definition 10 denotes that if activity \( a \) and \( b \) occur in executable traces of the model, there are at least \( k \)-activities between them. For instances, in Figure 14 \( BD_{\text{min}}(E, H, S) = 2 \), \( BD_{\text{min}}(A, I, S) = 3 \).

Theorem 2 Let \( L \) be an event log over \( \mathcal{A} \) and \( S = (N, M_\theta) \) be a net system, where \( N = (P, T, F, \lambda) \) is a labeled Petri net. \( \forall a, b \in \mathcal{A} \), if \( BD_{\text{min}}(a, b, L) = 1 \), \( BD_{\text{min}}(a, b, S) > 1 \) and \( a \not\Rightarrow b \), then there exists a silent transition of skip type between \( a \) and \( b \).

Proof. We assumed that there is no silent transition of skip type between \( a \) and \( b \). As \( a \not\Rightarrow b \), there exists a directed path consisting of places and transitions from \( a \) to \( b \). In addition, \( BD_{\text{min}}(a, b, S) > 1 \) also holds. Hence, there is at least one visible transition between \( a \) and \( b \), so \( \forall \sigma \in \mathcal{T}(S) \) such that \( BD_{\text{min}}(a, b, \sigma) = 1 \), i.e., \( BD_{\text{min}}(a, b, L) = 1 \) is not correct, which obviously contradicts the condition. As a consequence, Theorem 2 holds.

Definition 11 (Pre - and Post - Transition Sets)

Let \( S = (N, M_\theta) \) be a net system, where \( N = (P, T, F, \lambda) \) is a labeled Petri net. \( \forall t \in T \), \( \hat{t}(t) = \{ t' \mid t' \in T \land (t')^* \cap t = \emptyset \} \), \( \hat{t}(t)_r \) is called a pre-transition set of \( t \). \( (t)^* \) is called a post-transition set of \( t \), which is defined similarly.

Definition 12(Model-based Minimum Behavior Distance of Concurrent Structure)

Let \( S = (N, M_\theta) \) be a net system, where \( N = (P, T, F, \lambda) \) is a labeled Petri net. \( t_1, t_2 \in T \), \( t_1 \) is an and-split transition, \( t_2 \) is an and-join transition. The set of all executable transition sequences of \( S \) is denoted as \( T(S) \). Model-based minimum behavior distance of concurrent structure between \( t_1 \) and \( t_2 \) is defined as:
\[ BD_{\text{min}}^1(t_1, t_2, S) = \{ k \mid \forall \sigma \in T(S) \wedge BD_{\text{dis}}(t_1, t_2, \sigma) \geq k \} \]  \tag{5}

According to Definition 12, we can get \( BD_{\text{dis}}^1(f, i, S) = 1 \) in Figure 18(b) and \( BD_{\text{dis}}^1(f, j, S) = 2 \) in Figure 18(d).

Theorem 2 cannot solve the silent transition in concurrent structures. Theorem 3 presents a method for mining silent transitions in concurrent structures to solve this limitation.

**Theorem 3** Let \( I \) be an event log over \( \mathcal{A} \) and \( S = (N, M_0) \) be a net system, where \( N := (P, T, F, \lambda) \) be a labeled Petri net. \( \mathcal{A} \cup \tau \) is all transition labels. \( \forall t_1, t_2 \in T \), where \( t_1 \) is an and-split transition, \( t_2 \) is an and-join transition. There exists a silent transition of skip type on concurrent branches between \( t_1 \) and \( t_2 \), if one of the following conditions holds:

1. \( BD_{\text{dis}}^1(\lambda(t_1), \lambda(t_2), L) < BD_{\text{dis}}^1(t_1, t_2, S) \), where \( \lambda(t) \neq \tau, \lambda(t_2) \neq \tau \) \tag{6}
2. \( \exists \sigma \in L, a \in \sigma \wedge b \in \sigma \wedge BD_{\text{dis}}^1(c, d, \sigma) < BD_{\text{dis}}^1(t_1, t_2, S) \), where \( \lambda(t) = \lambda(t_2) = \tau \).

Proof. Without loss of generality, only the concurrency structure of the most basic case is given to prove the correctness of the theorem. Let us assume that there exists no silent transition of skip type. We discuss it in two cases:

1. when \( t_1 \) and \( t_2 \) are both visible transitions, the corresponding possible substructure is shown in Figure 15. \( \forall \sigma \in T(S) \), if \( t_1 \in \sigma \), then \( \sigma \) is the form of \( \cdots t_1 a t_2 b \cdots \) or \( \cdots t_1 b a t_2 \cdots \). Clearly, \( BD_{\text{dis}}^1(t_1, t_2, S) = 3 \). If there is no silent transition on the concurrent structure between \( t_1 \) and \( t_2 \), then \( BD_{\text{dis}}^1(\lambda(t_1), \lambda(t_2), L) \geq 3 \) that contradicts the conditions of \( BD_{\text{dis}}^1(t_1, t_2, S) \).

2. when \( t_1 \) and \( t_2 \) are both silent transitions, the corresponding possible substructure is shown in Figure 16. \( \forall \sigma \in T(S) \), if \( a \in \sigma \wedge b \in \sigma \), then an executable trace is the form of \( \lambda(t_1) = \lambda(t_2) = \tau \). \( \forall \sigma \in L \), if \( \exists \sigma \in L, a \in \sigma \wedge b \in \sigma \), then \( BD_{\text{dis}}^1(c, d, \sigma) \geq 3 \). Therefore, if there is no silent transition on the concurrent structure between \( t_1 \) and \( t_2 \), it is impossible to make \( BD_{\text{dis}}^1(c, d, \sigma) < BD_{\text{dis}}^1(t_1, t_2, S) \) hold. As a consequence, Theorem 3 holds.

Based on Algorithm 1, Algorithm 2 presents a method to discover silent transitions of skip type. The basic idea of Algorithm 2 is to compare the minimum behavior distance of activities in the log with their minimum behavior distance in the model, and determine whether there is a silent transition of skip type between them using Theorem 2 and Theorem 3.

**Algorithm 2** A method to discover silent transitions of skip type

```
1: M \rightarrow \text{An event log }\log \text{ and a Petri net model }M_0 \text{ obtained by Algorithm 1, the behavior distance matrix }BD_{\text{disMatrix}}^1
2: Output: \text{An Petri net with multiple types of silent transitions }M_1
3: traverse : BD_{\text{disMatrix}}^1
4: foreach a, b \in \mathcal{A} do
5: 1. if \( BD_{\text{disMatrix}}^1(a, b, L) \neq 1 \) then
6: Go to value \( BD_{\text{disMatrix}}^1(a, b, S) \)
7: 2. if \( BD_{\text{disMatrix}}^1(a, b, S) = 1 \) then
8: InsertSilent(\( a, b, \text{skip-type} )) // Insert a silent transition of type skip between a and b
9: end
10: end
11: foreach transitions of and-split and and-join do
12: let \( t_1 \) be an and-split transition, \( t_2 \) be an and-join transition
13: if \( \lambda(t_1) \neq \lambda(t_2) \) \( \neq \lambda(t_1) \) \( \neq \lambda(t_2) \) then
14: Go to value \( BD_{\text{disMatrix}}^1(t_1, t_2, L) \)
15: InsertSilent(\( t_1, t_2, \text{skip-type} )) // Insert silent transition inside a concurrent branch between \( t_1 \) and \( t_2 \)
16: end
17: end
18: return A Petri net with multiple types of silent transitions
```

For the initial Petri net model \( M_0 \) in Figure 13, according to Step 2-9 of Algorithm 2, we can find a silent transition of skip type between activity pairs \((J, M)\). The silent transition \( t_1 \) between \( A \) and \( I \) can be discovered using step 18-23 of Algorithm 2. Finally, an optimized Petri net \( M_1 \) with multiple types of silent transitions is obtained, as shown in Figure 17.

**VII.EXPERIMENTS**

In this section, series of experiments with real and synthetic event logs have been conducted to assess the goodness of our approach. Section A presents the evaluation criteria used in this paper. Section B presents the synthetic event logs used in this section. In Section C, we compare the proposed method with existing approaches in terms of the
number of silent transitions discovered and the quality of the process model using various synthetic event logs. In Section D, we discuss the experimental result on real event logs.

A. EVALUATION CRITERIA

Determining the quality of a process mining result is characterized by many dimensions. Usually, four main quality dimensions are used: fitness, precision, simplicity, generalization. In this paper, we use two main and most important quality criteria: fitness, precision. Fitness is to measure how much behavior in the event log can be replayed by the model. A model with good fitness allows for the behavior seen in the event log. The model has a perfect fitness if all traces in the log can be replayed by the model from beginning to end. There are various ways of defining fitness. Here, we use align-based technique to compute the fitness of the discovered model [24].

Precision is to measure how much behavior the model produces that is not observed in the event log. Here, we use the method proposed in [25] to calculate the precision. The model having a low precision will lead to the underfitting phenomenon. Therefore, a good mining algorithm should score well on the precision of the process model as can as possible when the fitness is good. Sometimes, putting aside a small amount of behavior causes a slight decrease in the fitness value, whereas the precision value increases much more. Therefore, we use the F-measure that combines fitness and precision as follows [26].

\[
F\text{–measure} = \frac{2 \times \text{Precision} \times \text{Fitness}}{\text{Precision} + \text{Fitness}}
\]  

Rediscovery mainly concerns whether the discovered process model is equivalent to the original model. When comparing two models, we only consider the behavioral relationship between them in this paper. However, the traditional trace equivalence yields a true or false answer and can, therefore, not be directly applied if models overlap partially. It cannot intuitively reflect what extent behavioral equivalence is between two models. Therefore, this paper quantifies the degree of consistency between the mined model and the original model using the trace consistency measurement proposed in literature [23].

B. SYNTHETIC EVENT LOGS

The event logs used in this section are mainly from two aspects. One is the previous synthetic event log from Section 2, and the other is generated by simulating a known process model, which contains multiple silent transitions of different types. For real event logs, there is no reference process model available to compare with the results of process discovery algorithms. Moreover, the correct number of silent transitions in the process model is also uncertain. Therefore, we designed fifteen artificial process models with different behavior and different types of silent transitions. The maximum number of activities in one process model is less than 20. 15 groups of event logs are generated by simulating fifteen artificial process models. Therefore, unlike real event logs, for each synthetic event log there is a corresponding reference process model. We can use these reference models as the ground truths that indicate how many silent transitions exist in the original models. Parts of the models are shown in Figure 18. The first model contains silent transitions of skip type, the second one contains silent transitions of loop and skip type, the third model contains silent transitions of side, and-gateway and skip type, the fourth one contains many silent transitions of skip type and loop, the fifth process model contains silent transitions of loop, and-gateway and skip type, and the last one contains all the previous silent transitions.
G. Evaluation Results with Synthetic Event Logs

For the previous synthetic event log and the 15 groups of event logs generated artificially, the number of silent transitions, the fitness of the model [21], the precision [22], and the consistency between the discovered model and the original model [23] are compared respectively.

The comparison results based on the event log $L_4$ are depicted in Table 1. The experimental results indicate that the proposed approach is more superior than others in terms of the number of silent transitions, the fitness and the precision of the model.

| Name | The proposed approach | $\alpha#$ algorithm | IM algorithm |
|------|-----------------------|---------------------|--------------|
| The number of silent transitions | 5 | 3 | 12 |
| fitness | 1 | 0.975 | 1 |
| precision | 1 | 1 | 0.962 |

We apply IM algorithm, $\alpha#$, and the proposed method on the 15 groups of event logs generated artificially and give a comparison result to evaluate the proposed method. For the event log generated by model 1 shown in Figure 18(a), when applying the proposed method and $\alpha#$ algorithm on this event log, three silent transitions in the original model can be rediscovered. Whereas IM algorithm produces four silent transitions, it discovers two silent transitions on the path from activity B to activity D instead of one silent transition. Therefore, the process model constructed by IM algorithm will have more additional behaviors. For the event log generated by model 2 shown in Figure 18(b), three silent transitions can be found using the proposed method and $\alpha#$ algorithm. However, the difference is that $\alpha#$ algorithm inserts a silent transition between the pre-set place and post-set place of $G$, thus the discovered model generates an additional executable sequence in the form of ‘... fhi...’.

Five silent transitions with the redundant silent transition of and-type are discovered by IM algorithm. For the event log generated by model 3 shown in Figure 18(c), the proposed method and IM algorithm both can rediscover four silent transitions included in the original model. Meanwhile, the model discovered by them has a perfect fitness and precision. However, the $\alpha#$ algorithm only discovers one silent transition of side-type and the fitness of the discovered process is less than 1. For the event log generated by model 4 shown in Figure 18(d), the proposed method discovers 4 silent transitions included in the original model. The $\alpha#$ algorithm discovers 3 silent transitions and fails to discover the silent transition occurred in the concurrent structure, which leads to the fitness less than 1. However, the model discovered by IM algorithm produces 14 silent transitions, which leads to the precision less than 1. However, for the event logs generated by Figures 18(a), 18(b), 18(c) and 18(d), the proposed method can all rediscover the original model and have perfect fitness and precision. For the event log generated by model 5 shown in Figure 18(e), the proposed method can discovered 6 silent transitions, however the $\alpha#$ algorithm can only discover 1 silent transition, which leads to the fitness and precision less than 1. However, IM algorithm and the proposed method discover 7 silent transitions and has a perfect fitness and precision. For the event log generated by model 6 shown in Figure 18(f), the proposed method discovers 8 silent transitions, $\alpha#$ algorithm can only discover 4 silent transitions which leads to the precision and fitness are both less than 1. However the model obtained by IM algorithm includes 5 silent transitions, which leads to the fitness less than 1.

Figure 19(a), 19(b), 19(c), 19(d) and 19(e) show the comparison results of IM algorithm, $\alpha#$ algorithm, and the proposed method in terms of the number of silent transitions, the fitness, the precision, the consistency and F-measure. Figure 19(a) indicates that the method proposed in this paper can identify multiple types of silent transitions in the original model. The number of silent transitions lies between the number of the IM algorithm and the $\alpha#$ algorithm without too many redundant silent transitions. Figures 19(b) and 19(c) indicate that the proposed method may cause a little decrease in fitness value in some situation, yet always yield a notable increase in precision value compare with others. Figure 19(d) shows that compared with the IM algorithm and $\alpha#$ algorithm, the behavior of the model obtained by this method is more consistent with that of the original model. Figure 19(e) shows that the F-measure of the obtained process models by the proposed approach has a better result than the others. In conclusion, the experimental results show that the proposed method can find multiple types of silent transitions. Meanwhile, it improves the quality of process discovery results.
D. Evaluation Using Real Event Data

In this section, we discuss the result using the real logs from the Business Process Intelligence Challenge (BPIC), which are accessible via 4tu Center for Research Data. Table 2 reports the characteristics of all logs in terms of number of case, number of events, number of unique labels, minimum number of events per trace, maximum number of events per trace for each log. We mined the process models with three different discovery algorithms, the proposed approach, IM algorithm, α# algorithm used 4 BPIC logs. Results of this experiment are given in Table 3.

Table 2: Basic Information of These Event Logs

| Event log      | #Case | #Event | #activity | #min-event-per-trace | #max-event-per-trace |
|----------------|-------|--------|-----------|----------------------|----------------------|
| BPIC_2012      | 13087 | 262200 | 36        | 3                    | 175                  |
| BPIC_2013_incidents | 7554  | 65533  | 13        | 1                    | 123                  |
| BPIC_2013_open | 819   | 2351   | 5         | 1                    | 22                   |
| BPIC_2014_close| 1487  | 6660   | 7         | 1                    | 35                   |

Table 3: The Experiment Results Applying Different Real-Life Event Logs and Discovery Technique

| Process Discovery | Metric     | BPIC_2012 | BPIC_2013_incidents | BPIC_2013_open | BPIC_2014_close |
|-------------------|------------|-----------|---------------------|---------------|-----------------|
|                   | Fitness    | 0.845     | 0.897               | 0.325         | 0.564           |
|                   | Precision  | 0.127     | 0.637               | 0.491         | 0.695           |
|                   | #Silent     | 6         | 4                   | 3             | 5               |
|                   | F-measure   | 0.225     | 0.739               | 0.809         | 0.512           |
|                   | Transition  | 0.847     | 0.816               | 0.916         | 0.864           |
|                   | Precision  | 0.798     | 0.869               | 0.679         | 0.344           |
|                   | #Silent     | 36        | 59                  | 22            | 22              |
|                   | F-measure   | 0.831     | 0.842               | 0.809         | 0.512           |
|                   | Transition  | 0.923     | 0.928               | 0.916         | 0.864           |
|                   | Precision  | 0.867     | 0.875               | 0.873         | 0.823           |
|                   | #Silent     | 16        | 26                  | 18            | 12              |
|                   | F-measure   | 0.894     | 0.903               | 0.894         | 0.843           |
|                   | Transition  | 16        | 26                  | 18            | 12              |

Figure 20 shows the result of silent transition, obtained when applying three techniques on different real-life event logs. As we see, for these four real event logs, the proposed method produces fewer redundant silent transitions than the IM algorithm. Moreover, the number of silent transitions mined by the proposed method lies between the other two algorithms.

Figure 21 shows the obtained fitness, precision, F-measure values of applying different methods on real event logs. For BPIC_2012 and BPIC_2013_Open event log, the process model obtained by α# algorithm has perfect fitness. However, the precision of the model is poor as it contains several isolated activities which can produce arbitrary numbers.

---

1. https://data.4tu.nl/articles/dataset/BPI_Challenge_2012/12689204
   https://data.4tu.nl/articles/dataset/BPI_Challenge_2013_incidents/12693914
   https://data.4tu.nl/articles/dataset/BPI_Challenge_2013_open_problems/12688556
   https://data.4tu.nl/articles/dataset/BPI_Challenge_2013_closed_problems/12714476
behaviors. For BPIC_2013_Open and BPIC_2013_Close event log, the process model constructed by IM algorithm both has perfect fitness, but due to the existence of redundant transitions, loops, and concurrency, its precision is very poor, especially BPIC_2013_Close event log, which can produce executable sequences of any length. For BPIC_2013_Incidents event log, the model generated by the IM algorithm contains 50 silent transitions, including a large number of redundant silent transitions. Compared with the IM algorithm, the process model obtained by the proposed method achieves more high fitness and precision, as well as reduces the number of redundant silent transitions. For BPIC-2012 event log, the process model generated by α# algorithm contains a large number of isolated activities, that resulting in very low precision of the model. Although the process models built by other algorithms have perfect fitness, however the precision is very poor because of these models produce too many extra behaviors not in event log. Hence, the quality of the obtained process model is still relatively poor. In general, it is more reasonable to use F-measure to measure the quality of the process model. As shown in Figure 21, for all event logs, the proposed method can discover a process model with a high F-measure value.

Based on the experiments using real event log, we conclude that the proposed method presents a little drop in fitness and a significant increase in precision. However, the F-measure significantly improves when our technique is used compared the others. This significant increment of F-measure is explained by the noticeable and significant increment of precision. Therefore, the result from synthetic event logs and real-life event logs both confirm the effectiveness of our technique.

VIII. CONCLUSION AND FUTURE WORK
The paper proposes a novel method to mine process models with silent transitions based on behavior distance. First, to more accurately capture the dependencies between activities, the minimum and maximum behavior distances of activities based on the event log are calculated. And then, the basic behavior relationship between pairs of activities is obtained through the weak order relationship and behavior distance between activities. After that, an initial process model with silent transitions of and-gateway type and loop type is constructed by analyzing the basic behavior relationship and behavior distance. Based thereon, silent transitions of skip type are mined using behavior distance between activities based on log, model, and concurrent structure. Finally, lots of experiments have been done to verify the effectiveness of the proposed method. The experimental results show that the proposed method can successfully discover various types of silent transitions and improve the model’s F-measure without significantly reducing the fitness of the model. The future work mainly has two aspects: (1) considering how to improve the method to extend it to a non-free choice structure. (2) When the behavioral relationship of the log is incompleteness, how to correctly discover the possible behavioral relationship between activities.

REFERENCES
[1]. Van Der Aalst W. Process mining: discovery, conformance and enhancement of business processes[M]. Heidelberg: Springer, 2011.
[2]. Leemans S J J, Fahland D, van der Aalst W M P. Discovering block-structured process models from event logs-a constructive approach[C]//International conference on applications and theory of Petri nets and concurrency.
Algorithms for Discovering a Process Model Containing Invisible Task and Non-Free Choice

Sarno R, Sungkono K R, Johanes R, et al. Graph-Based Innovative Computing, Information and Control, 2018, 14(4).

Sarno R, Sungkono K R. Constructing Control-Flow Patterns Containing Invisible Task and Non-Free Choice Tasks[J]. Procedia Computer Science, 2017: 134-141.

Van der Aalst W M P, Weijters A J M M. Process mining: a research agenda[J]. Computers in Industry, 2004, 53(3): 231-244

Liu C, Duan H, Zeng Q, et al. Towards comprehensive support for privacy preservation cross-organization business process mining[J]. IEEE Transactions on Services Computing, 2019, 12(4): 639-653.

Liu C, Zeng Q, Cheng L, et al. Privacy-preserving behavioral correctness verification of cross-organizational workflow with task synchronization patterns[J]. IEEE Transactions on Automation Science and Engineering, 2021, 18(3): 1037-1048.

Liu C, Pei Y, Cheng L, et al. Sampling business process event logs using graph-based ranking model[J]. Concurrency and Computation: Practice and Experience, 2021, 33(5): 1-14.

Liu C, Zeng Q, Cheng L, et al. Measuring Similarity for Data-Aware Business Processes[J]. IEEE Transactions on Automation Science and Engineering, 2021: 1-13.

Wen L, Wang J, Sun J. Mining Invisible Tasks from Event Logs[C]// Advances in Data and Web Management, Joint 9th Asia-Pacific Web Conference, APWeb 2007, and 8th International Conference, on Web-Age Information Management, WAIM 2007, Proceedings. Springer-Verlag, 2007: 358-365.

Wen L, Wang J, Der Aalst W M, et al. Mining process models with prime invisible tasks[J]. Data & Knowledge Engineering, 2010, 69(10): 999-1021.

Aalst W M P V D, Medeiros A K A D, Weijters A J M M. Genetic Process Mining[C]// International Conference on Application and Theory of Petri Nets. Springer, Berlin, Heidelberg, 2005: 48-69.

X.Q. Huang, L.F. Wang, W. Zhao, S.K. Zhang, and C.Y. Yuan. A workflow process mining algorithm based on synchro-net[J]. Journal of Computer Science and Technology, 2006, 21(1): 66-71.

Rozinat A, van der Aalst W M P. Decision mining in ProM[C]// International Conference on Business Process Management. Springer, Berlin, Heidelberg, 2006: 420-425.

Guo Q, Wen L, Wang J, et al. Mining Invisible Tasks in Non-free-choice Constructs[C]// 13th International Conference on Business Process Management, BPM 2015. Springer, Berlin, Heidelberg, 2015: 109-125.

Sarno R, Sungkono K R. Coupled Silent Markov Model for Process Mining of Invisible Prime Tasks[J]. International Review on Computers and Software, 2016, 11(6): 539-547.

Sarno R, Sungkono K R. Coupled Silent Markov Model for Process Discovery of Non-Free Choice and Invisible Prime Tasks[J]. Procedia Computer Science, 2017: 134-141.

K. R. Sungkono and R. Sarno, Constructing Control-Flow Patterns Containing Invisible Task and Non-Free Choice Based on Declarative Model[J]. International Journal of Innovative Computing, Information and Control, 2018, 14(4).

Sarno R, Sungkono K R, Johanes R, et al. Graph-Based Algorithms for Discovering a Process Model Containing Invisible Tasks[J]. International Journal of Intelligent Engineering and Systems, 2019, 12(2): 85-94.

T. Murata. Petri nets: Properties, analysis and applications[J]. Proceedings of the IEEE, 1989, 77(4): 541-580.

Polyvyanyy A, Der Aalst W M, Hofstede A H, et al. Impact-Driven Process Model Repair[J]. ACM Transactions on Software Engineering and Methodology, 2017, 25(4)

Weidlich M, Polyvyanyy A, Desai N, et al. Process compliance analysis based on behavioural profiles[J]. Information Systems, 2011, 36(7): 1009-1025.

Weidlich M, Mendling J, Weske M, et al. Efficient Consistency Measurement Based on Behavioral Profiles of Process Models[J]. IEEE Transactions on Software Engineering, 2011, 37(3): 410-429.

Van der Aalst W, Adriansyah A, van Dongen B. Replaying history on process models for conformance checking and performance analysis[J]. Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery, 2012, 2(2): 182-192.

Rozinat, A, Van Der Aalst, W.M.P. . Conformance Testing: Measuring the Fit and Appropriateness of Event Logs and Process Models[C]// BPM 2005 International Workshops, BPI, BPD, ENEI, BPRM, WSCOBPM, BPS ,Springer Berlin Heidelberg, 2006: 163-176.

De Weerdt J, De Backer M, Vanthienen J, et al. A robust F-measure for evaluating discovered process models[C]// 2011 IEEE Symposium on Computational Intelligence and Data Mining (CIDM). IEEE, 2011: 148-155.