IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received 7 June 2023, accepted 21 June 2023, date of publication 27 June 2023, date of current version 3 July 2023.
Digital Object Identifier 10.1109/ACCESS.2023.3289876

RESEARCH ARTICLE

Behavior Differentiation of Process Variants With Invisible Tasks

JUAN GUO AND HUAN FANG
School of Mathematics and Big Data, Anhui University of Science and Technology, Huainan 232001, China
Corresponding author: Huan Fang (fanghuan0307@163.com)

This work was supported in part by the National Natural Science Foundation, China, under Grant 61902002.

ABSTRACT It is a fact that plenty of process variants are derived from the same base model in practical applications, and the main reasons include inevitable software maintenance and adaptability of process models. This fact raises a question that many process variants are with high similarity degree, and can not be differentiated from each other under the state-of-the-art similarity measurements. In order to differentiate similar-but-different process variants, we propose an approach of process variants' behavior differentiation in this paper. The method analyzes the complex structures of process variants with invisible tasks, and utilizes task execution relationship to construct an integrated similarity measurement, which extends the existing similarity measurement capacity in terms of dealing with invisible tasks. It is proved that the proposed task execution relationship can capture the dominate features of similar-but-different process variant including invisible tasks, which means that as long as the task execution behaviors of two process variants are different, the corresponding behavior matrices must be different. Furthermore, a set of experiments are carried out, in order to evaluate the properties of effectiveness, semantic uniqueness expression and correctness of the proposed method. Meanwhile, the experimental results give evidences that the proposed method outperforms the existing model similarity measurements in accuracy.

INDEX TERMS Behavior differentiation, behavior profiles, change mining, invisible task, silent transition, petri nets.

I. INTRODUCTION

In the course of increasing individualization of customer demand, configurable process models can offer various benefits like reusability and more flexibility compared with traditional predefined models. Usually, there exist many process models generated from the same base model, and these process models are with high similarity. Here, we call a set of process models as similar-but-different process variants [1], [2], which are derived through various configurable operations from a common base model.

In this setting, process variants are a subset of executions of a business process that can be distinguished from others based on some characteristics [3]. For example, an organization may have different process orchestrations for some given specific business processes, such as multiple products sales processes in different countries (say $C1$, $C2$, $C3$, and $C4$), or multiple accounting processes in different branches (say $B1$, $B2$, $B3$, and $B4$). So, the actual executions of the same process may vary with time and geography, and we can obtain four similar-but-different process variants: one for each of these countries or branches. In these variants, some relevant event data such as location, different business modules, products, and customer types could change, but the main process models are similar, and can be divided into different clusters. The sub-models of clusters are functionally homogeneous, but can be differentiated from each other by some partial variations, and these similar models can be formalized, understood and expressed as process variants.

Process variant analysis is a family of techniques to analyze the models and event logs produced during the actual execution of models, in order to identify and explain the differences between two or more process variants. A wide range of methods for process variants analysis have been proposed in the

The associate editor coordinating the review of this manuscript and approving it for publication was Zhiwu Li.

last decade, such as configurable Business Process Modelling Notation (BPMN), Petri nets [1], [3], configurable process mining [4], etc. The goal of process variant analysis is to help business analysts to understand why and how multiple process variants differ.

In principle, process executions are expressed as activities in BPMN, while in Petri nets they are represented as transitions [5]. Generally speaking, in the models of process variants, multiple transitions may bear the same label. One can think of the transition label as the observable action. Sometimes one wants to express those specific transitions as unobservable. For this, we reserve the label ε, and a transition t with the label ε is unobservable, denoted as $l(t) = \varepsilon$. Such unobservable transitions are often referred to as silent transitions or invisible tasks. Although invisible tasks are not explicitly recorded in the event log, they need to be reflected in the model, in other words, silent transitions or invisible tasks can limit the expressive power of the model [5].

Process similarity techniques have proven to be an effective solution of managing large process repositories [6], [7]. The essence of process similarity metrics is to analyze the degree of similarity between two process models given different business objectives and to quantify this degree into specific similarity values. At present, most of the existing methods of process model behavior similarity are based on the execution timing relationship between tasks in the process model [7], [8], [9], which could bring about the problem that they can not deal with some complex structures, such as structures with invisible tasks. To solve the problem, this paper proposes an integrated similarity measurement to evaluate behavior differences among different variants with invisible tasks, and presents a process similarity measurement approach based on Task Execution Relationship (TER), and highlights the execution relationship between tasks based on the order of the tasks in the process model. Furthermore, we propose a behavior differentiation method of process variants based on TER matrix.

The main contributions of this paper are as follows:

(1) The construction of TER matrix to express the behavior of process models based on task execution relationships, which takes invisible tasks into consideration. It is proved that there exists a one-to-one mapping between TER matrix and process models, in other words, if two process models have different behaviors, the corresponding TER matrices are also different.

(2) The construction of a behavior differentiation method for process variants based on TER similarity measurement. It is shown that there exist multiple process variants with the similarity degree of 1 by using the state-of-the-art techniques, which means that it is impossible to tell these process variants apart using the current similarity techniques. However, by utilizing the proposed behavior differentiation method for process variants, we can differentiate different process variants whose models or event logs demonstrate behavior differences. Meanwhile, we verify the proposed method, and confirm its accuracy through a set of benchmark datasets.

In the following parts, some related work is discussed in Section II, and the problem statement is given in Section III. Some basic knowledge about the behavior differentiation of process variant models is clarified in Section IV. In Section V, the proposed behavior differentiation method is proposed, which utilizes the similarity measurement for process models and takes invisible tasks into consideration. In Section VI, we conduct experiments and analyze the experimental results. Finally, conclusions and suggestions for future work are presented in Section VII.

II. RELATED WORK

In this section, we review some related work about behavior differentiation of process variants and process similarity techniques.

A. BEHAVIOR DIFFERENTIATION OF PROCESS VARIANTS

Process variants usually have similar-but-different model structures, so it is obvious that the model structures are the essential tools to differentiate process variants. However, in some scenarios, the models of process variants can not be known, and there can be seen actual execution event logs only. So, in total there are two kinds of behavior differentiation techniques with the first kind starting from process models, and the second from event logs directly. Here, we call the former techniques as process variability modelling methods, and the latter configurable process mining methods.

1) PROCESS VARIABILITY MODELLING METHODS

Given the base model of a specific business process as known, some process variants clusters or families can be obtained via configuring personalized operations to the base model [10], [11]. These configurable operations can be performed by stakeholders, managers, or end-users, and can be expressed by multiple forms, such as Not-Functional-Requirement [12], reasonable process fragments [13], and declarative variability rules [14].

Although the digital and the physical worlds are closely aligned, and it is possible to track operational management processes in detail to some extent, there exist challenges to differentiate these process variants [15]. While employing model-based comparisons to multiple process variants, the key problem is related to the fact that the variants are compared in terms of their structures whereas we aim to compare the behaviors. Thus, a kind of low-level behavioral representation is preferred, instead of high-level process modelling languages, such as BPMN or Petri nets [5]. However, low-level modelling methods such as transition systems fall short in state-explosion. Moreover, the existing process variability modelling methods are mostly from the perspective of control flow, and another drawback of model-based approaches is that they are unable to detect differences in terms of traces frequencies or other perspectives. Therefore, some additional comprehensive techniques could be taken into consideration, supporting advanced process variability modelling.

2) CONFIGURABLE PROCESS MINING METHODS

This kind of process variants mining methods starts from event logs directly, and does not depend on any priori knowledge about the business process model [3], [16]. The first step is mostly splitting the event logs into cohorts using some trace merging, splitting operations, process fragmentation, and event log slicing [17]. Owing to the fact that process variants are composed of several process fragments with commonalities and differences, these similarities can be used to merge a cluster of variants together. The work in [3] formalizes the concepts of configurable process models and their variability in change mining field, and proposes an approach of merging and filtering a collection of event logs from the same family with respect to variability. This kind of method aims to enhance change mining from a collection of event logs and detect changes in variable fragments of the obtained event log.

The state-of-the-art techniques can discover related process variants in an interactive manner, by utilizing the information of event logs, such as frequency of trace, control flow relationship, and even performance indicators. This method usually generates pattern collections of WF-nets, and each workflow pattern describes a trace cluster, which can be further used to discover process variants. To address the problem that existing works in configurable process discovery lack the incorporation of the semantics in the resulting model, the work in [4] reports a method to enrich the collection of event logs with configurable process ontology concepts by introducing semantic annotations, which can capture variability of elements present in the event log.

It is noteworthy that configurable process mining may lead to the generation of incomprehensible "spaghetti like" process models, especially in some highly flexible environments. One of the main reasons is the diversity of event logs, for example, there are local, non-significant differences between several process execution instances. Therefore, in order to solve the problems caused by local diversity of event logs, tracing clustering methods can be used [18], [19]. Tracing clustering techniques divide the logs into more homogeneous subsets by reducing the number of process log instances that participate in the analysis at one time. The obtained research conclusions show that trace clustering techniques will enhance process mining [19]. However, to the best knowledge of us, there exists no evidence showing that trace clustering techniques could enhance configurable process mining for process variants.

B. PROCESS SIMILARITY MEASUREMENTS

Over the past few decades, a series of research achievements have been made on process similarity measurements. The existing studies on process model similarity measurement mainly include the following four aspects: (1) textual similarity of process models [20]; (2) structural similarity of process models [21], [22]; (3) execution semantics of process models [23], [24], [25], [26], [27], [28], [29], [30]; (4) combining process model structures and execution logs together [20], [31], [32], [32], [33]. Compared with the textual similarity of process models and structural similarity, the actual execution logs of process models can describe the dynamic behavior semantics of a process model comprehensively, and hence reflect the essence of a process model. Therefore, process similarity measurement based on behavior semantics has become a hot topic in the related research field in recent years.

TABLE 1. The types of business model applicable to similarity techniques.

Similarity technique	Referencing source	Applicable model type
BP	[15]	WF-net
BP^+	[26]	WF-net
CBP	[27]	WF-net
ETR	[33]	WF-net
TAR	[24]	Unrestricted
TOR	[25]	Unrestricted
CF	[28]	WF-net
Our work	this paper	WF-net

The textual similarity mainly refers to the label textual information similarity of the elements contained in the process models. The structural similarity mainly reflects the similarity of the process model topology which expresses the logical relationship between business activities. The behavioral similarity emphasizes the execution semantics of business models. The similarity measures combine the log behavior with the model structure, and take process actual preference into consideration. Therefore, we argue that the last kind of similarity measurement can better reflect the execution preference that facilitates recommending similar processes with high accuracy.

Typical similarity measurement techniques include TAR [24], BP [15], CF [28], ETR [33], TOR [25], BP^+ [26], and these techniques have laid down a solid foundation for process models in model inquiries, or recommendations. Besides, each similarity technique has its own applicable business model, which is concluded and shown in Table 1. Obviously, as described in Table 1, the most applicable model of these extant similarity techniques is WorkFlow Net (WF-net for short).

A Workflow net is a special type of Petri nets. Specifically, a workflow net is a Petri net with a dedicated source place where the process starts and a dedicated sink place where the process ends, and all nodes are on a path from source to sink [5]. Hereafter, for the sake of brevity, a workflow net is shorten as a WF-net.

C. BUSINESS PROCESS DEVIANCE MINING

The behavior differentiation research in this paper is similar to those in the field of business process mining to some extent. Business process deviance mining is the research area that aims to characterize deviations of a business process from its expected outcomes [34]. Existing techniques for business process deviance mining are based on the extraction of patterns from event logs, using different pattern mining approaches [34], [35], [36], [37], [38], [39], [40], [41], [42].

Given event logs of a business system as known, process discovery methods construct a process model representing the processes recorded in the log, while conformance checking techniques quantify how well the discovered model achieves this objective. Conformance checking techniques can be used to detect, locate and explain deviations, and to measure the severity of these deviations. Techniques within conformance checking discover which features of a set of process executions are associated with changes, such as possible correlations among process activities, frequencies of events, control-flow information, and multi-perspective data flow information [36], [37], [40], [41]. Process deviance mining provides insights on which process leads to the best performance and also reveals behaviors that result in undesired process outcomes. Besides, most of the traditional conformance checking techniques are off-line; however, on-line or multi-perspective conformance checking techniques are highlights of the current research.

Although our work in this paper is similar to process deviance mining study, our work doesn't belong to it. The process deviance mining techniques usually take the derivation of a process model from its actual execution event logs, whereas our work focuses on the behavior differentiation among a set of similar-but-different process variants, and in particular, these process variants are expressed as Petri nets, and include invisible tasks or silent transitions.

D. SUMMARY OF RELATED WORK

As outlined in the previous subsections, a wide range of methods and techniques have been proposed to tackle the problem of process variants analysis and behavior differentiation. The comparisons between our work and those in literature review are concluded in Table 2.

From the perspective of considering complex model structures with invisible tasks, this paper proposes an approach of behavior differentiation for process variants, which could have a high similarity degree under the existing techniques. The proposed method highlights the tackling of four types of invisible tasks, and presents a framework of behavior differentiation based on task execution relationships.

III. PROBLEM STATEMENT

In order to describe the main target of this work, and provide a better understanding of the deficiencies of the existing process similarity measurement techniques, especially when they are applied to evaluate the behavior differentiation among process variants, we first give an example to illustrate the motivation of this work in this section.

There is a cluster of process variants, including three process models named WFN_1, WFN_2 and WFN_3 respectively, expressed by labelled WF-net modeled by Petri nets, as shown in Fig. 1. WF-nets are a kind of business process models, where there exists a unique entrance place, a unique exit place, and no dead nodes. The definition of WF-nets will be given in next section. As shown in Fig. 1, the black filled

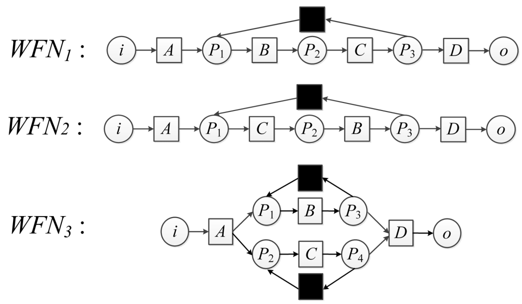


FIGURE 1. Three labelled WF-net systems with silent transitions.

rectangle represents silent transition (invisible task), whose transition label is ε.

We utilize the current four process similarity measurement methods to evaluate the similarities between any two models in Fig. 1. The selected similarity measurement methods are ETR [33], BP [15], CBP [27], and TOR [25] respectively, and the process similarity indices are calculated and shown in Table 3.

It can be observed that by using the existing four kinds of similarity measurements, the similarity of any two of the process models in Fig. 1 is equal to 1 (shown in Table 3). In other words, the above depicted three process models in Fig. 1 are considered to be the same, under the state-of-the-art similarity measurements. However, there exists distinct behavior differentiation among the three process models in Fig. 1. For example, activity A in WFN_1 can be directly followed by activity B, in WFN_2 activity A can be directly followed by activity C, while in WFN_3 activity A can be directly followed by activity B or activity C. So, from the perspective of task execution semantics, WFN_1, WFN_2 and WFN_3 are different from each other; however, the four similarity measurements cannot tell them apart, as the similarity degrees of "1" are obtained, which is inaccurate obviously.

IV. PRELIMINARIES

In this paper, we use WF-net as a formal modeling and analysis tool, representing different business processes and measuring the similarity between them. Some basic concepts are presented in this section.

The Petri net model was first proposed by Carl Adam Petri [43], which is suitable for describing asynchronous concurrent operations, and has been widely used in many fields such as intelligent control [43], [44], [45], business system modelling [36], etc.

A Petri net [32] is a triplet $N = (P, T, F)$, where P is a nonempty finite set of places, T is a nonempty finite set of transitions, $F \subseteq (P \times T) \cup (T \times P)$ is a set of arcs, and $P \cap T = \emptyset$. In a net N, the preset and postset of a node $x \in P \cup T$ are defined as ${}^\bullet x = \{y \mid y \in P \cup T \wedge (y, x) \in F\}$ and $x^\bullet = \{y \mid y \in P \cup T \wedge (x, y) \in F\}$, respectively.

This notation can be extended to a set of nodes. For any node set X ($X \subseteq P \cup T$), (1) and (2) are defined as the preset

TABLE 2. Comparisons of our work with the existing techniques.

Techniques	Prerequisite	Objectives
Process variability modelling methods	process models	Adopting Variable and configurable operations to process models.
Configurable process mining methods	event logs	Mining log fragment to demonstrate semantic variability captured from event log.
Process similarity measurements	process models with or without event logs	Evaluating the similarity of process models.
Process deviation detection methods	process models and event logs	Detecting deviation of process model from actual event logs.
Our work	process models	Differentiating similar-but-different behaviors among process variants.

TABLE 3. The similarity measurement results of different algorithms for the process model in Fig. 1.

Algorithm	Result of similarity measurement		
models	WFN_1 and WFN_2	WFN_1 and WFN_3	WFN_2 and WFN_3
ETR	1.00	1.00	1.00
BP	1.00	1.00	1.00
CBP	1.00	1.00	1.00
TOR	1.00	1.00	1.00

and postset of X, respectively.

$$^{\bullet}X = \bigcup_{x \in X} {}^{\bullet}x \tag{1}$$

$$X^{\bullet} = \bigcup_{x \in X} x^{\bullet} \tag{2}$$

Given a place p, $\mathcal{B}(p)$ denotes the number of tokens in p. A marking M is the set of all multisets over the places set P. $\Sigma = (N, M)$ is called a marked net, where M is the initial marking of Petri net N.

In a marked net (N, M), a transition $t \in T$ is enabled if all places p in the preset of t satisfy $M(p) \geq 1$, which is denoted as $(N, M)[t\rangle$. A marking M' is directly reachable from M when t fires, which is denoted as $(N, M)[t\rangle M'$, satisfying $M' = M + t^{\bullet} - {}^{\bullet}t$.

If there exists a transition firing sequence $\sigma = t_0, t_1, \cdots, t_{n-1}$ of length $n \in \mathbb{N}$, which satisfies $(N, M)[t_0\rangle M_0, M_0[t_1\rangle M_1, \cdots, M_{n-2}[t_{n-1}\rangle M_{n-1}$ and $M_{n-1} = M'$, then M' is reachable from M in the net system (N, M), i.e., $M' \in R(N, M)$.

Definition 1: **(Boundness, Safeness and Livenss) [32]**

Let $\Sigma = (P, T, F, M_0)$ be a marked Petri net, where M_0 is the initial marking. $\Sigma = (P, T, F, M_0)$ is said to be bounded, if for each place $p \in P$, there exists a natural number $n \in \mathbb{N}$ such that for each reachable marking $M \in RM(N, M_0)$, the number of tokens in p is less than or equal to n.

$\Sigma = (P, T, F, M_0)$ is said to be safe, if and only if for each place p under all markings $M \in RM(N, M_0)$ implies that $M(p) \leq 1$.

$\Sigma = (P, T, F, M_0)$ is said to be live, if and only if for each reachable marking $M \in RM(N, M_0)$ and each transition $t \in T$, there exists a marking $M' \in RM(N, M_0)$ reachable from M, which enables t.

Definition 2: **(WF-net system) [32]**

A marked net $WFN = (P, T, F, M_0)$ that satisfies the following conditions is called a workflow net system.

(1) There exists a unique source place $i \in P$ satisfying that ${}^{\bullet}i = \emptyset$;

(2) There exists a unique sink place $o \in P$ satisfying that $o^{\bullet} = \emptyset$;

(3) $N = (P, T, F)$ is strong connected Petri net. For each node $x \in P \cup T$, x is in a directed path from i to o;

(4) $M_0 = [i]$ is the initial marking of Petri net $N = (P, T, F)$.

Definition 3: **(Soundness) [32]**

Let $WFN = (P, T, F, M_0)$ be a WF-net system with $M_0 = [i]$. WFN is said to be sound if the following conditions hold.

(1) Proper completion: for any marking $M \in RM(M_0)$, $o \in M$ implies $M = [o]$;

(2) Option to complete: for any marking $M \in RM(M_0)$, $[o] \in RM(M)$;

(3) Absence of dead transitions: $WFN = (P, T, F, M_0)$ contains no dead transitions, in other words, for any $t \in T$, there is a firing sequence enabling t.

Obviously, a WF-net system $WFN = (P, T, F, M_0)$ has the only initial marking $[i]$ and the only final marking $[o]$, and each of them has only one token in place i or place o, respectively.

Definition 4: **(Path)**

In a WF-net system $WFN = (P, T, F, M_0)$, node sequence $\langle n_1, n_2, \cdots, n_k\rangle$ is called a path if the following condition is satisfied:

$$\forall i \in [1, k-1] : n_i \in P \cup T \wedge (n_i, n_{i+1}) \in F \tag{3}$$

Based on the concept of path, precursor, common precursor and nearest common precursors are defined as follows.

Definition 5: **(Precursor, Common Precursor, and Nearest Common Precursor)**

Let $x \in P \cup T$ be a node in a WF-net system $WFN = (P, T, F, M_0)$, and t be a transition in T.

(1) If there exists a path from x to t, node x is a precursor of transition t, denoted as $x \in pre(t)$.

(2) If $x \in pre(t_1) \cap pre(t_2)$, x is called the common precursor of transitions t_1 and t_2, denoted as $x \in cmpre(t_1, t_2)$.

(3) If $x \in cmpre(t_1, t_2)$, and there exists no node x' such that $x' \in cmpre(t_1, t_2) \wedge x \neq x' \wedge x \in pre(x')$, x is called the nearest common precursor of transitions t_1 and t_2, denoted as $x \in ncpre(t_1, t_2)$.

Definition 6: **(Full Firing Sequence, *FFS*)**

Let $WFN = (P, T, F, M_0)$ be a WF-net system with $M_0 = [i]$. Marking $[i]$ means the only initial marking in the source place i, marking $[o]$ means the only final marking in the sink place o, and there is only one token in place i or place o. A transition sequence σ_i is called a full firing sequence if $[i]\sigma_i\rangle[o]$ holds.

The set of all full firing sequence is denoted as FFS, i.e., $FFS = \{\sigma_i \mid i \geq 0\}$.

Definition 7: **(Labelled WF-net)**

A five-tuple $LWFN = (P, T, F, \Phi, L, M_0)$ that satisfies the following conditions is called a labelled WF-net:

(1) $WFN = (P, T, F, M_0)$ be a WF-net system;

(2) Φ is the set of activity label for transitions;

(3) $L : T \rightarrow \Phi \cup \{\varepsilon\}$ is a function of assigning labels to transitions, and ε denotes empty label for silent transitions.

Definition 8: **(Task Execution Relationship, TER)**

Let $LWFN = (P, T, F, \Phi, L, M_0)$ be a labelled WF-net, and $x, y \in T$ be two transitions in $LWFN$. $TER = \{\rightarrow, \rightsquigarrow, \infty, NAN\}$ is called as the task execution relationship between x and y, denoted as $R(x, y) \in TER$, if the following conditions are satisfied.

(1) If there exists a transition sequence $\sigma = \langle t_1, t_2 \cdots t_n\rangle \in FFS$ $(i, j \in \{1, 2, \cdots, n\}, 1 \leq i \leq n-1)$ that satisfies $t_i = x \wedge t_{i+1} = y \wedge L(t_i) \neq \varepsilon \wedge L(t_{i+1}) \neq \varepsilon$, then transitions x and y are in direct follow relationship, denoted as $x \rightarrow y$.

(2) If there exists a transition sequence σ $\langle t_1, t_2 \cdots t_n\rangle \in FFS$ $(i, j \in \{1, 2, \cdots, n\}, 1 \leq i < j-1 \leq n)$ that satisfies $t_i = x \wedge t_j = y \wedge L(t_i) \neq \varepsilon \wedge L(t_j) \neq \varepsilon$, then transitions x and y are in indirect follow relationship, denoted as $x \rightsquigarrow y$.

(3) If transitions x and y are not in relationship $x \rightarrow y$ or $x \rightsquigarrow y$, and $L(x) \neq \varepsilon \wedge L(y) \neq \varepsilon$, then transitions x and y are in unreachable relationship, denoted as $x \infty y$.

(4) If transitions x and y satisfy $L(x) = \varepsilon \vee L(y) = \varepsilon$, then transitions x and y are in a NAN value relationship, denoted as x NAN y.

Definition 9: **(TER matrix, *TERM*)**

Let $LWFN = (P, T, F, \Phi, L, M_0)$ be a labelled WF-net. $TERM(LWFN)$ is called as the TER matrix of $LWFN$, if the following two conditions are satisfied.

(1) The size of TER matrix is $\mid T \mid \times \mid T \mid$, where $\mid T \mid$ means the number of transition in T, including invisible transitions.

(2)

$$TERM(LWFN)_{i,j} = R(t_i, t_j) \in TER \tag{4}$$

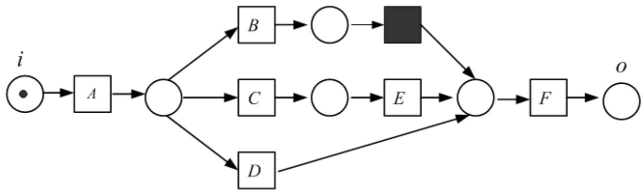


FIGURE 2. A labelled WF-net system Σ_1 with silent transition.

TABLE 4. The TER matrix of Σ_1.

	A	B	C	D	E	F
A	∞	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightsquigarrow$	$\rightsquigarrow$
B	∞	∞	∞	∞	∞	$\rightarrow$
C	∞	∞	∞	∞	$\rightarrow$	$\rightsquigarrow$
D	∞	∞	∞	∞	∞	$\rightarrow$
E	∞	∞	∞	∞	∞	$\rightarrow$
F	∞	∞	∞	∞	∞	∞

For the sake of brevity, all NAN values are omitted in $TERM$ matrix.

The labelled WF-net system Σ_1 shown in Fig. 2 is taken as an example to illustrate the concept of task execution relationship. The full firing sequence FFS of Σ_1 is denoted as $FFS = \{ABF, ACEF, ADF\}$, and its corresponding $TERM$ matrix is shown in Table 4.

V. CONSTRUCTION OF BEHAVIOR DIFFERENTIATION METHOD FOR PROCESS VARIANTS

A. PROPOSED BEHAVIOR DIFFERENTIATION METHOD FRAMEWORK

In order to differentiate behaviors of similar-but-different process variants, we propose a framework of process similarity measurement based on TER matrix, and further expand the research conclusions of the existing process similarity measurement methods. And the biggest innovation lies in the fact that it takes the TER of any pair of tasks including invisible tasks into consideration.

The main procedures are as follows.

(1) Calculating the nearest common precursor for any pair of tasks in the process model;

(2) Computing TER matrix $TERM$;

(3) Performing additional handling on structures containing invisible tasks, and updating $TERM$ of the process models;

(4) Calculating similarity of process variants based on $TERM$, which can tell the behaviors of processes apart.

The specific algorithms for these four steps are described in the following subsections.

B. NEAREST COMMON PRECURSOR COMPUTING ALGORITHM

The nearest common precursor of two tasks can reflect the execution relationship between them, so it can be used to differentiate the types of invisible tasks. Algorithm 1 calculates the corresponding nearest common precursor for two given tasks, and the corresponding pseudo code is shown below.

Algorithm 1 Nearest common precursor computing algorithm (*computeNCP*(x, y) for short.)

Require: Tasks set T of process model, and two tasks x and y.
Ensure: The nearest common precursor $ncpre(x, y)$ of tasks x and y.
1: Initialize a queue $Q = \emptyset$;
2: Initialize a set $S = \emptyset$;
3: **for** each $z \in^{\bullet} x$ **do**
4: $enqueue(Q, z)$;
5: **while** $notempty(Q)$ **do**
6: $q = del(Q)$;
7: **if** $q \in^{\bullet} y$ **then** then
8: $S = S \cup \{q\}$;
9: **end if**
10: **end while**
11: $x = z$, goto step 3;
12: **end for**
13: **return** $ncpre(x, y) = S \cap T$.

We utilize a queue Q and a set S to extract the nearest common precursor of tasks x and y, and initialize Q and S firstly (lines 1 and 2). Then, we enqueue z in Q, where z is the preset of task x (lines 3 and 4). When Q is not empty, the front of Q is named q: if q is the preset of task y, then q is added to set S (lines 5–10). Thereafter, we assign z to x to complete iteration (lines 11 and 12). Finally, line 13 returns the result of the nearest common precursor $ncpre(x, y)$.

C. TER MATRIX CALCULATION ALGORITHM

Calculation steps of the values in TER matrix are shown in Algorithm 2, where each value denotes the task execution relationship between any two tasks in the corresponding WF-net.

According to Definitions 8 and 9, lines 1–7 of Algorithm 2 calculate all the direct follow relationship in T, and lines 8–14 calculate all the indirect follow relationship in T. Besides, according to the result of $ComputeNCP(x, y)$ in algorithm 1, the related direct follow relationship is obtained in lines 15–21. Otherwise, the relationships among the retained tasks are set to ∞ (lines 22–28), indicating that there exists neither direct follow relationship, nor indirect follow relationship. Finally, $TERM$ is returned in line 29.

D. HANDLING OF INVISIBLE TASKS

Invisible tasks refer to tasks that exist in the process model but do not appear in the actual execution log. In other words, invisible tasks correspond to the transitions with label ε in a labelled WF-net system, and are represented as black filled rectangles in the model. Most of the latest process model similarity measurement methods ignore invisible tasks. In fact,

Algorithm 2 TER matrix calculation algorithm

Require: A WF-net $WFN = (P, T; F, M_0)$.
Ensure: Task execution relation matrix $TERM$.
1: **for** each $p \in P \backslash \{i, o\}$ **do**
2: **for** each $x \in {}^{\bullet}p$ **do**
3: **for** each $y \in p^{\bullet}$ **do**
4: set $x \rightarrow y$ in $TERM$;
5: **end for**
6: **end for**
7: **end for**
8: **for** each $z \in T$ **do**
9: **for** each $x \in T$ **do**
10: **for** each $y \in T \wedge (x \rightarrow z \vee x \rightsquigarrow z) \wedge (z \rightarrow y \vee z \rightsquigarrow y) \wedge (x \nrightarrow y)$ **do**
11: set $(x \rightsquigarrow y)$ in $TERM$;
12: **end for**
13: **end for**
14: **end for**
15: **for** each $x \in T$ **do**
16: **for** each $y \in T$ **do**
17: **if** $computeNCP(x, y) = z$ and $x \neq z \wedge y \neq z$ **then**
18: set $(z \rightarrow x), (z \rightarrow y)$ in $TERM$;
19: **end if**
20: **end for**
21: **end for**
22: **for** each $x \in T$ **do**
23: **for** each $y \in T$ **do**
24: **if** the relation between x and y is not set before **then**
25: set $(x \infty y)$ in $TERM$;
26: **end if**
27: **end for**
28: **end for**
29: **return** $TERM$.

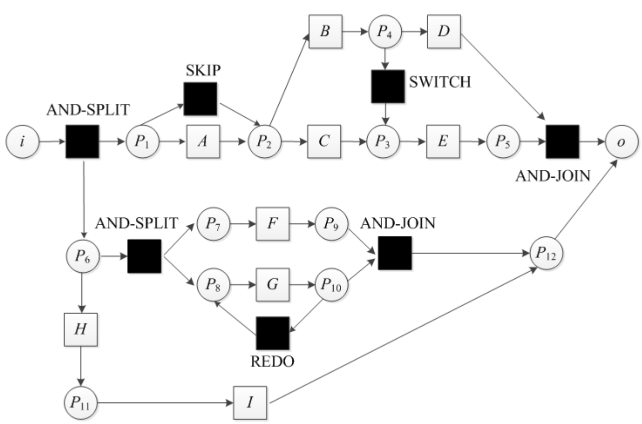


FIGURE 3. Illustration of different kinds of invisible tasks.

the previous section of problem statement also shows that ignoring the impact of invisible tasks may lead to inaccurate similarity measurement results. Therefore, this section introduces the differentiation method of various types of invisible tasks and its corresponding handling method.

As shown in Fig. 3, invisible tasks can be divided into four types, named *SKIP*, *REDO*, *SWITCH* and *AND* gateways, respectively. It is noteworthy that *AND* gateway includes not only *AND−SPLIT* and *AND−JOIN* gateways, but also *SIDE* gateways, because the function and role of *SIDE* gateway are the same as *AND* gateway [46].

With an invisible task τ, its corresponding preset and postset are denoted as P_1 and P_2, respectively. The type of τ can be determined by the following procedures:

(1) *AND* gateway can be located simply according to the number of the preset and postset of the invisible task. If both of $| P_1 |= 1$ and $| P_2 |\geq 1$ are satisfied, then x is called $AND - JOIN$ gateway.

(2) The determinations of *SKIP*, *REDO* and *SWITHCH* gateways are comparatively complex. As $| P_1 |$ and $| P_2 |$ are all equal to 1, so the type determination should be judged by the nearest common precursors, which can be calculated from the model that deleting the corresponding invisible task.

The type determination algorithm of invisible tasks is presented in Algorithm 3 for specifically.

Algorithm 3 Type determination algorithm for invisible tasks

Require: A WF-net *WFN* and an invisible task $\tau \in T$.
Ensure: Type τ of task $TYPE(\tau)$ is not visible.

1: **for** each τ **do**
2: **if** $|{}^{\bullet}\tau| = 1 \wedge |\tau^{\bullet}| > 1$ **then**
3: $TYPE(\tau) = AND - SPLIT$;
4: **else if** $|\tau^{\bullet} = 1| \wedge |{}^{\bullet}\tau| > 1$ **then**
5: $TYPE(\tau) = AND - JOIN$;
6: **else if** ${}^{\bullet}\tau = P_1$ and $\tau^{\bullet} = P_2 \wedge computeNCP\,(P_1, P_2) = P_1$ **then**
7: $TYPE(\tau) = SKIP$;
8: **else if** ${}^{\bullet}\tau = P_1$ and $\tau^{\bullet} = P_2 \wedge computeNCP\,(P_1, P_2) = P_2$ **then**
9: $TYPE(\tau) = REDO$;
10: **else if** ${}^{\bullet}\tau = P_1$ and $\tau^{\bullet} = P_2 \wedge computeNCP\,(P_1, P_2) = P_3$ **then**
11: $TYPE(\tau) = SWITCH$;
12: **end if**
13: **end for**
14: **return** $TYPE(\tau)$.

For a given invisible task τ, if $|{}^{\bullet}\tau| = 1 \wedge |\tau^{\bullet}| > 1$, then the type of τ is called $AND-SPLIT$, as shown in lines 2 and 3 in algorithm 3; similarly, if $|\tau^{\bullet}| = 1 \wedge |{}^{\bullet}\tau| > 1$, then the type of τ is called $AND - JOIN$ (lines 4 and 5). Likewise, *SKIP*, *REDO*, and *SWITCH* can also be obtained (lines 6–13).

The executable codes corresponding to Algorithm 1–3 are available online[1], which are implemented by the Python language.

[1] https://github.com/zhangzhengzhuifeng/algorithms

After the type of invisible tasks in the process model is determined by Algorithm 3, different invisible tasks can be handled accordingly. As shown in the following cases, this paper deals with four types of invisible tasks.

(1) **Case 1:** Given τ is an invisible task, if $type(\tau) \in \{AND, SKIP, REDO, SWITCH\}$, then invisible task τ is treated as a special kind of visible tasks, when calculating the task execution relationship matrix *TERM*. Specifically, if a visible tasks named t satisfies that $ComputeNCP(\tau, t) \neq \emptyset$, then the value of $TERM_{(\tau,t)}$ is set as ∞ in *TERM*.

(2) **Case 2:** Given τ is an invisible task, if $type(\tau) \in \{AND, REDO, SWITCH\}$, then it is set that $x \rightarrow y$, where $x \in^{\bullet} \tau \wedge y \in \tau^{\bullet} \wedge x \rightsquigarrow y$. For the reason that invisible tasks would not appear in the execution sequence of the process model, these kinds of gateway type only play the role of routing in the process model, and no further special handling operation is required.

(3) **Case 3:** Given τ is an invisible task, if $type(\tau) \in \{SKIP\}$, then it is necessary to check each path between task τ and task y, where $y \rightsquigarrow \tau \vee \tau \rightsquigarrow y$, due to the particularity of invisible tasks of *SKIP* type. After completion of handling in Case 2, if there exists a task execution relationship $a \rightsquigarrow b$, and the path from task a to task b contains no visible task, then the task execution relationship between a and b is replaced by $a \rightarrow b$; otherwise, the task execution relationship of a and b will not be updated.

(4) **Case 4:** Since invisible tasks of *SKIP* type are involved in filling in the task execution relationship matrix, they need to be marked through a special set of tags to distinguish different invisible tasks of *SKIP* types. The principle of naming rules for special tags is to use *SKIP* as the prefix of the tag, and use the tag of visible tasks whose scope is skipped as the suffix in the matrix. The tags in the suffix are arranged in dictionary order.

Invisible tasks are processed according to the above handling procedures, and the task execution relationship matrix *TERM* can be obtained. For example, the *TERM* of the process model in Fig. 3 is obtained, as shown in Table 5.

E. BUILDING FRAMEWORK OF SIMILARITY MEASUREMENT USING TASK EXECUTION RELATIONSHIP

As mentioned above, when the task execution relationship matrix is obtained, the similarity between two process models based on *TERM* is successively determined. There are many researches on the task alignment in process models, and this content is beyond the scope of this paper. We assume that if the tasks in two process models have the same label, then they refer to the same task.

Definition 10: **(Similarity between two process models based on *TERM*)**

Let $WFN_1 = (P_1, T_1, F_1, \Phi_1, L_1, M_{01})$, $WFN_2 = (P_2, T_2, F_2, \Phi_2, L_2, M_{02})$ be two labelled WF-net systems, and $TERM_1$, $TERM_2$ be the task execution relationship matrix of WFN_1 and WFM_2, respectively, $r_1, r_2 \in \Gamma = \{\rightarrow, \rightsquigarrow, \infty\}$. sim $(WFN_1, W\,FN_2)$ is called the similarity between

TABLE 5. ***TERM*** **containing each type of invisible task model.**

	A	B	C	D	E	F	G	H	I	$SKIP(A)$
A	∞	$\rightarrow$	$\rightarrow$	$\rightsquigarrow$	$\rightsquigarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	∞
B	∞	∞	∞	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	NAN
C	∞	∞	∞	∞	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	NAN
D	∞	∞	∞	∞	∞	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	NAN
E	∞	∞	∞	∞	∞	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	NAN
F	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	∞	$\rightarrow$	∞	∞	NAN
G	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	∞	∞	NAN
H	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	∞	∞	∞	$\rightarrow$	NAN
I	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	∞	∞	∞	∞	∞
$SKIP(A)$	∞	$\rightarrow$	$\rightarrow$	$\rightsquigarrow$	$\rightsquigarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	$\rightarrow$	∞

TABLE 6. **The similarity weight parameter setting in this work.**

w_{r_1,r_2}	$\rightarrow$	$\rightsquigarrow$	∞
$\rightarrow$	1.0	0.6	0.0
$\rightsquigarrow$	0.6	1.0	0.0
∞	0.0	0.0	1.0

WFN_1 and WFM_2, and it is calculated by (5) and (6).

$$\mathrm{sim}\,(WFN_1, W\ FN_2) = \frac{\mid TERM_1 \cap TERM_2 \mid}{\mid TERM_1 \mid + \mid TERM_2 \mid - \mid TERM_1 \cap TERM_2 \mid} \tag{5}$$

$$\mid TERM_1 \cap TERM_2 \mid = \sum_{x_1 r_1 y_2 \in TERM_1 \wedge m(x_1) r_2 m(y_2) \in TERM_2} w_{r_1,r_2} \tag{6}$$

In (5) and (6), $m(x)$ is a task alignment function for task x, in other words, $m(x)$ means the alignment task of x in counterpart process model, and w_{r_1,r_2} is the similarity weight between different task execution relationships.

It is noteworthy that w_{r_1,r_2} is a user-defined parameter. In the subsequent similarity calculation, the value of the similarity weight between different task execution relationships in this paper is shown in Table 6.

So far, the introduction of similarity calculation method based on task execution relationship is completed, and in next section several experiments are carried out to evaluate the performance of the proposed method.

VI. EVALUATION OF PROPOSED METHOD

For the sake of convenience, we refer to the proposed similarity measurement method based on *TERM* as TER method for short.

The experimental evaluation of this work is mainly divided into four stages. The first stage compares the effectiveness of different similarity measurement methods through six manually constructed process models. The second stage conforms the uniqueness of the behavior semantics of TER method by constructing process models. The third stage checks the correctness of TER method through a set of benchmark dataset that consists of 100 process models [47][2]. And the fourth stage verifies the accuracy of TER method through a set of benchmark dataset.

[2]https://github.com/wjxSky/Benchmark-Dataset/tree/master/Benchmark

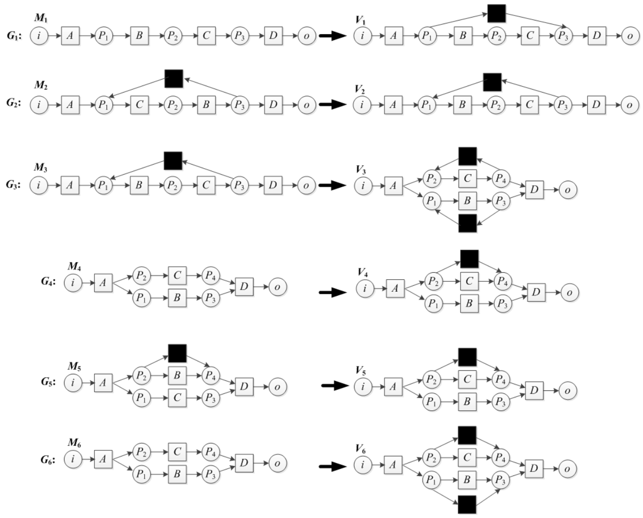


FIGURE 4. **Six groups of base models and their counterpart variants.**

A. EXPERIMENT SETTING

This section mainly verifies the processing ability of TER method to differentiate some complex process model structures. The experimental data include six groups of 12 process models, each of which is composed of a base model and its process variant obtained by operating some changes to the structure of base model. The corresponding grouping information is shown in Table 7.

Six behavior similarity measurement methods of process models are compared with the proposed TER method in this paper. The specific comparison methods are shown in Table 8.

The six manually constructed process models are shown in Fig. 4.

B. EFFECTIVENESS ANALYSIS OF EXPERIMENTAL RESULTS

Seven similarity measurement methods in Table 8 are taken to calculate the similarity of Groups G_1–G_6 process models, and the obtained results are shown in Table 9.

As we know, two process models are considered to be the same if the similarity degree between them equals one. As for the process variants of G_1–G_6, it is obvious that there exists a specific base model, and its variants are regarded as having the same behavior. For example, M_1 and V_1 in G_1 are regarded as the same under the method of A_2 or A_3 (shown in Table 9). However, it can be deduced that the behaviors in M_1 and V_1 in

TABLE 7. Experimental grouping information.

grouping	The base model	Changes in operating	Process variant model
G_1	M_1	Adding invisible tasks of *SKIP* type	V_1
G_2	M_2	Changing the execution order of tasks within the loop structure	V_2
G_3	M_3	Changing the sequential structure within the cyclic structure to the parallel structure	V_3
G_4	M_4	Adding invisible tasks of *SKIP* type to a branch of the parallel structure	V_4
G_5	M_5	Adding invisible tasks of *SKIP* type to counterpart branches of the parallel structure	V_5
G_6	M_6	Adding invisible tasks of *SKIP* type to all branches of the parallel structure	V_6

TABLE 8. Information of comparison methods.

Method number	Method name	Literature sources
A_1	TAR	[24]
A_2	ETR	[33]
A_3	BP	[15]
A_4	CBP	[27]
A_5	TOR	[25]
A_6	BP^+	[26]
A_7	TER	this work

TABLE 9. Effectiveness experimental results.

Method name	Similarity of base model and its variants in different groups					
	G_1	G_2	G_3	G_4	G_5	G_6
A_1	0.750	0.333	0.500	1.000	1.000	0.857
A_2	1.000	1.000	0.947	1.000	1.000	1.000
A_3	1.000	1.000	1.000	1.000	1.000	1.000
A_4	0.857	1.000	1.000	0.780	0.857	0.820
A_5	0.900	1.000	1.000	0.900	0.820	0.820
A_6	0.951	0.818	0.568	0.975	0.975	0.951
A_7	0.765	0.778	0.818	0.800	0.666	0.951

G_1 are obviously different, as shown in Fig. 4. Therefore, the proposed TER similarity measurement can differentiate all similar-but-different process variants, for the reason that the similarity degrees under the proposed TER method in Table 9 are not equal to the value of 1. The lower the similarity value, the greater the degree of difference in behavior. Both the BP^+ method and our proposed TER method can distinguish similar-but-different behaviors in process variant groups of G_1–G_6; however, the BP^+ method does not consider the silent transitions of the model, while our method does.

C. ANALYSIS AND COMPARISON OF EXPERIMENTAL RESULTS

The effectiveness of the process similarity measurement method lies in the fact that it can distinguish two process models with high structural similarity but different behaviors. Here, in order to verify the effectiveness of different process similarity methods, 12 process models and their variants are constructed in Fig. 4, and the results are shown in Table 9. To sum up, a comprehensive effectiveness comparison is shown in Table 10, where ✓ denotes that the method can distinguish the similar process structure, while $\times$ denotes that it cannot distinguish.

It is obvious that the extant similarity methods have some deficiencies, such as A_1–A_5. There are some erroneous similarity results, as there are some "1"s in Table 9, in other words, A_1–A_5 methods can not tell similar process variants apart. Our proposed method (A_7) and BP^+ (A_6) can distinguish similar behaviors of these process variants, as the similarity degrees measured by A_6 and A_7 are less than 1. BP^+ method is effective and efficient, and it is based on important properties pertaining to uniqueness, span and necessity & sufficiency [26]. Compared with the BP^+ method, by revealing the flaws of the state-of-the-art similarity techniques in dealing with invisible tasks, our proposed method highlights the capacity of invisible tasks handling. The TER method proposed in this paper cannot only correctly and effectively deal with the process model including cyclic structure and invisible tasks, but construct a one-to-one matching cooperation between the TER matrix and the process model. The semantic behavior uniqueness of the TER matrix will be discussed in the next subsection through experiments.

D. BEHAVIOR UNIQUENESS EXPRESSION VERIFICATION

The behavior uniqueness expression of TER method means that TER method can characterize the task execution

TABLE 10. The effects of different methods are compared.

groups	Changing operation	A_1	A_2	A_3	A_4	A_5	A_6	A_7
G_1	Adding invisible tasks of *SKIP* type	✓	×	×	✓	✓	✓	✓
G_2	Changing the execution order of tasks within the loop structure	✓	×	×	×	×	✓	✓
G_3	Changing the sequential structure to the parallel structure within the cyclic structure changes	✓	✓	×	×	×	✓	✓
G_4	Adding invisible tasks of *SKIP* type to a branch of the parallel structure	×	×	×	✓	✓	✓	✓
G_5	Adding invisible tasks of *SKIP* type to counterpart branches of the parallel structure	×	×	×	✓	✓	✓	✓
G_6	Adding invisible tasks of *SKIP* type to all branches of the parallel structure	✓	×	×	✓	✓	✓	✓

relationship between any pairs of task a in process model. In other words, the generated task execution relationship matrix behaves uniquely for any process model. As long as the behaviors of the two models P and Q are different, their corresponding TER matrices obtained must be different, and $sim(P, Q) < 1$. Similar to the uniqueness verification of *ExRORU* algorithm [20], the uniqueness verification of TER method in this work is also based on the construction method.

By adding an arbitrary one-step change operation to a free choice WF-net named P, a new process model named Q (also a free choice WF-net) is obtained. Here, the mentioned free choice WF-net is a special kind of WF-net, in which for any two tasks $t_1, t_2 \in T$, $t_1 \neq t_2 \wedge {}^\bullet t_1 \cap {}^\bullet t_2 \neq \emptyset$ implies that $|{}^\bullet t_1| = |{}^\bullet t_2| = 1$. It can be proved that the TER matrices of P and Q are different from each other, starting from three perspectives by graphical demonstrations as the following procedures under various scenarios.

Theorem 1: Let us assume that P, Q are two free choice WF-nets, and Q is generated through continuous one-step change operations to P. If P and Q have different behaviors, their corresponding TER matrices are different from each other.

Proof: This proposition is proved by constructive proof method, through the following scenarios.

(1) Scenario 1: Changing the relationship of a pair of tasks.

If the relationship of a pair of tasks is changed, for example, a pair of tasks in the sequence structure are transformed into a parallel structure or a cyclic structure, and this kind of structural change will have a direct impact on the TER matrix of the process model according to Definition 8.

(2) Scenario 2: Adding or deleting a visible task.

When adding a new visible task to the existing process model structure, this added visible task must be in sequential relationship, parallel relationship, mutual exclusion relationship, or cyclic relationship with an existent task. Fig. 5 depicts several process model structures generated by possible addition of visible task in different locations. For the sake of clarity, only part of the changing structure of a WF-net is shown in Fig. 5. In Fig. 5, B is a visible task to be inserted into the model, and S represents the precursor task immediately before the changing structure, and E represents the successor tasks immediately after the changing structure.

It can be concluded that an obvious change will occur to the TER matrix of process model after taking any single step of visible task adding operation, as shown in Fig. 5. Likewise, similar proofs can be used to the analysis of deleting a visible task, and it is omitted for the sake of brevity.

Due to the fact that a new process variant model is eventually constructed by continuously applying the above single step changes (adding or deleting) to the original process model, it can be concluded that the new process model and the original process model have different TER matrices.

(3) Scenario 3: Adding or deleting an invisible task.

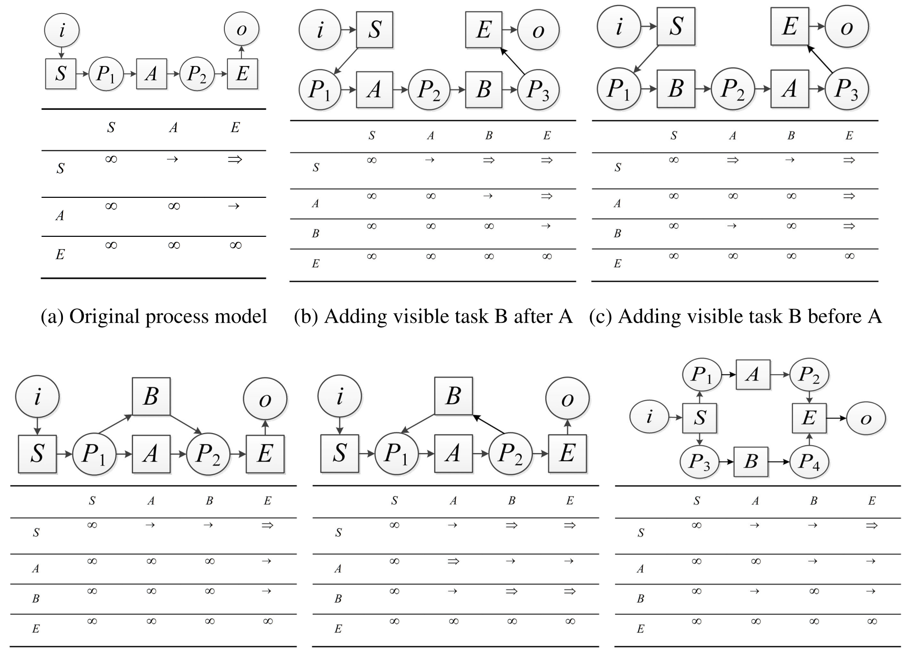
A new invisible task can also be added to or deleted from the original process model, which is similar to the structures in Scenario 2. The process models and their corresponding TER matrices of adding an invisible task are shown in Fig. 6.

It can be seen from Fig. 6 and the task execution relationship matrix of each process model that any single step change of adding invisible tasks will cause the corresponding TER matrix to change. Likewise, similar proofs can be used to the analysis of deleting an invisible task.

In summary, the proposition is proved. ■

Compared with the existing process model similarity methods, the proposed TER method has the advantages of effectively handling process model similarity involving invisible tasks, and it can distinguish the behavior differences caused by sequential, cyclic and parallel structures.

From the above-mentioned experiments, it is obvious that for any two sound free choice WF-nets P and Q, if P and Q show any different behavior on their traces, their corresponding TER matrices must be different. So, even if the models are restricted to the simplest WF-nets, such as free choice WF-nets, the latest process similarity measurement methods are incapable of depicting the semantic uniqueness characterization. In contrast, the proposed TER method can effectively



(a) Original process model (b) Adding visible task B after A (c) Adding visible task B before A

(d) Task B is mutually exclusive with task A (e) Task B and Task A form a cyclic structure (f) Task B is parallel to task A

FIGURE 5. Inserting a task into base model.

TABLE 11. The similarity values of 10 process models obtained by TER method.

	M_1	M_2	M_3	M_4	M_5	M_6	M_7	M_8	M_9	M_{10}
M_1	1.00	0.66	0.79	0.83	0.21	0.47	0.52	0.11	0.23	0.31
M_2	0.66	1.00	0.12	0.38	0.24	0.13	0.12	0.81	0.54	0.66
M_3	0.79	0.12	1.00	0.51	0.65	0.23	0.61	0.33	0.97	0.27
M_4	0.83	0.38	0.51	1.00	0.49	0.43	0.30	0.95	0.87	0.09
M_5	0.21	0.24	0.65	0.49	1.00	0.48	0.72	0.32	0.65	0.71
M_6	0.47	0.13	0.23	0.43	0.48	1.00	0.14	0.18	0.57	0.52
M_7	0.52	0.12	0.61	0.30	0.72	0.14	1.00	0.74	0.55	0.08
M_8	0.11	0.81	0.33	0.95	0.32	0.18	0.74	1.00	0.00	0.23
M_9	0.23	0.54	0.97	0.87	0.65	0.57	0.55	0.00	1.00	0.21
M_{10}	0.31	0.66	0.27	0.09	0.71	0.52	0.08	0.23	0.21	1.00

deal with invisible tasks, so as to distinguish the differences caused by sequential, cyclic and parallel structures.

E. CORRECTNESS EVALUATION

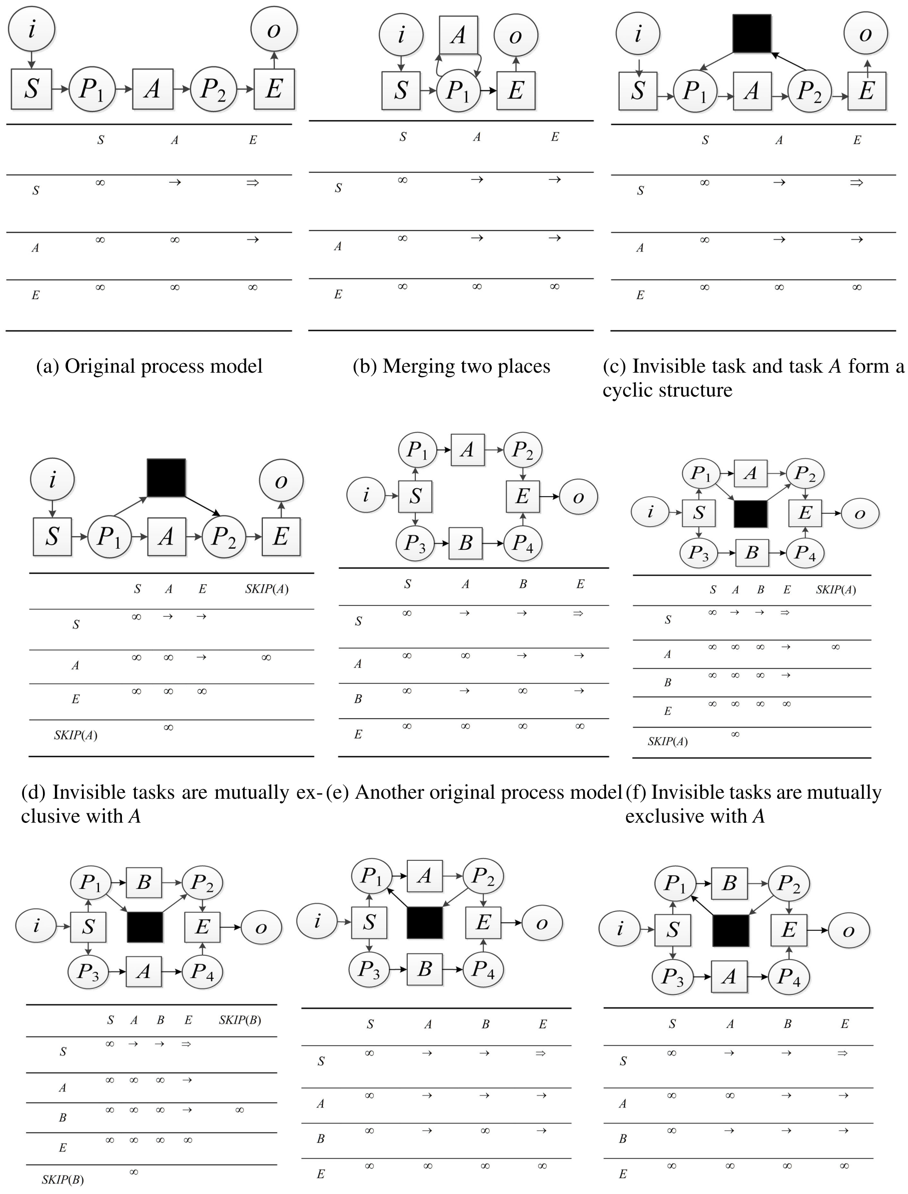
In order to evaluate the correctness of TER method, a benchmark dataset that consists of 100 process models is used [47], and randomly divided into 10 groups thereafter, that is, 10 for each group. Then, by using the proposed TER method in this paper, the similarity value between a pair of models in each group is calculated, and the results are listed in Table 11.

It can be seen that the experimental results obtained by TER method are all within the range of [0,1] in Table 11. When a pair of processes are identical, their corresponding similarity degree equals one; likewise, when a pair of processes are completely different, their corresponding similarity degree equals zero. The feasibility and correctness of the TER method have been verified in previous subsections.

F. ACCURACY EVALUATION OF TER METHOD

In terms of accuracy evaluation, we select a public set of benchmark data for the evaluation [47]. The data consist of 100 process models, among which 10 retrieval processes and their 9 related processes are marked, as well as the sorting order of related processes. For each retrieval process, the order of relevance between the 9 related processes and the retrieval process is determined by the results of user survey, and is used as a benchmark to evaluate accuracy of the TER method.

The accuracy evaluation of TER method is carried out according to the following steps. Firstly, we use the benchmark dataset and its related processes to calculate the process similarity based on TER method and the counterpart methods listed in Table 8; then, we sort the obtained similarity results, and compare the sorted ranking with the benchmark ranking; finally, we evaluate the accuracy of each method by applying Normalized Discount Cumulative Gain(*NDCG* for short), a family of ranking measures widely used in practice, which



(a) Original process model

	S	A	E
S	∞	→	⇒
A	∞	∞	→
E	∞	∞	∞

(b) Merging two places

	S	A	E
S	∞	→	→
A	∞	→	→
E	∞	∞	∞

(c) Invisible task and task A form a cyclic structure

	S	A	E
S	∞	→	⇒
A	∞	→	→
E	∞	∞	∞

(d) Invisible tasks are mutually exclusive with A

	S	A	E	SKIP(A)
S	∞	→	→	
A	∞	∞	→	∞
E	∞	∞	∞	
SKIP(A)		∞		

(e) Another original process model

	S	A	B	E
S	∞	→	→	⇒
A	∞	∞	→	→
B	∞	→	∞	→
E	∞	∞	∞	∞

(f) Invisible tasks are mutually exclusive with A

	S	A	B	E	SKIP(A)
S	∞	→	→	⇒	
A	∞	∞	∞	→	∞
B	∞	∞	∞	→	
E	∞	∞	∞	∞	
SKIP(A)		∞			

(g) Invisible task is mutually exclusive with B

	S	A	B	E	SKIP(B)
S	∞	→	→	⇒	
A	∞	∞	∞	→	
B	∞	∞	∞	→	∞
E	∞	∞	∞	∞	
SKIP(B)		∞			

(h) Invisible task and task A form a cyclic structure

	S	A	B	E
S	∞	→	→	⇒
A	∞	→	→	→
B	∞	→	∞	→
E	∞	∞	∞	∞

(i) Invisible task and task B form a cyclic structure

	S	A	B	E
S	∞	→	→	⇒
A	∞	∞	→	→
B	∞	→	→	→
E	∞	∞	∞	∞

FIGURE 6. Adding an invisible task to base model.

adopts a logarithmic discount that converges to one as the number of items to rank goes to infinity [48].

1) MEASUREMENT STANDARDS

The relevant formula of *NDCG* is shown as (7) and (8).

$$DCG_n = \begin{cases} r(n), n = 1 \\ r(1) + \sum_{n=2}^{n} \frac{r(n)}{\log_n 2}, n > 1 \end{cases} \tag{7}$$

$$NDCG_n = \frac{DCG_n}{IDCG_n} \times 100\% \tag{8}$$

In (7) and (8), DCG_n is the *NDCG* value of a specific query model and its n related processes, and $r(n)$ is the weight of the relevant process in the $n - th$ position, which is user-defined. $IDCG_n$ represents the ideal discount cumulative gain of processes under the standard sorting of n related processes, in other words, *IDCG* is the value of the *DCG* value under the ideal case of sorting. This accuracy experiment mainly explores the similarity values of 10 retrieval processes and their corresponding related processes, and the average accuracy of their ranking, namely, the average normalized discount cumulative gain (*ANDCG* for short). The higher the *ANDCG* value, the higher the archive accuracy of this method.

TABLE 12. Detailed information of retrieval processes.

process *id*	1	2	3	4	5	6	7	8	9	10
number of places	14	112	42	42	73	39	101	91	46	48
number of events	10	10	46	48	45	32	99	84	36	36
number of edges	26	222	92	96	144	76	220	198	96	100
structures	*S*	*S*	*E*	*E*	*P*	*P*	*EP*	*EP*	*EPL*	*EPL*

TABLE 13. Similarity ranking of search processes and related processes.

Group	retrieval process *id*	Relevant process number (in descending order according to the similarity value with the retrieval process)
1	1	2,3,4,5,6,7,8,9,10
2	11	12,13,14,15,16,17,18,19,20
3	21	22,23,24,25,26,27,28,29,30
4	31	32,33,34,35,36,37,38,39,40
5	41	42,43,44,45,46,47,48,49,50
6	51	52,53,54,55,56,57,58,59,60
7	61	62,63,64,65,66,67,68,69,70
8	71	72,73,74,75,76,77,78,79,80
9	81	82,83,84,85,86,87,88,89,90
10	91	92,93,94,95,96,97,98,99,100

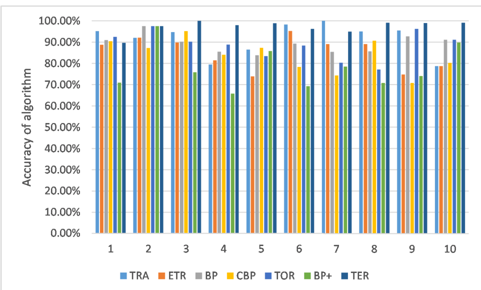


FIGURE 7. ***NDCG*** **values for different methods.**

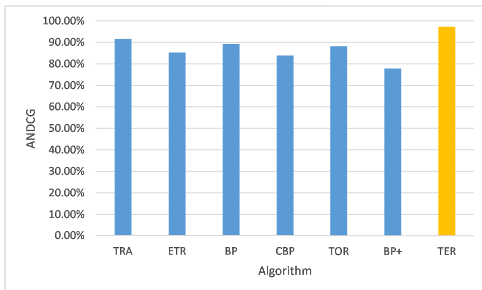


FIGURE 8. ***ANDCG*** **values for different methods.**

The structural details of the 10 retrieval processes in the experimental data are shown in Table 12, where *S*, *E*, *P*, *L*, *EP* and *EPL* represent the combination of sequential structure, selective structure, parallel structure, cyclic structure, sequential+selective structure and sequential+selective+cyclic structure respectively.

The data designer can make changes to each retrieval process by adding or deleting edges/nodes. For example, No. 1 retrieval process in Table 13 only contains a sequential structure; however, by adding or deleting edges/nodes to the No. 1 retrieval process, a related process can be obtained, which includes a selection, parallel, cyclic structure or different combinations of various structures. Therefore, the processes in the benchmark dataset basically cover all possible control flow changing structures. Table 13 shows the specific information about retrieval process dataset, such as group identifier, retrieval process model identifiers, and the relevant process models sorting information.

2) ACCURACY EXPERIMENT RESULTS ANALYSIS

For the 10 retrieval processes marked in the dataset, by calculating the values of the relevant processes and their sorting order corresponding to each retrieval process, the process archive accuracy results of investigated methods are obtained. Fig. 7 shows the *NDCG* values of each method, and Fig. 8 shows the *ANDCG* (Average *NDCG* among different processes) values of each method.

From Figs. 7 and 8, it can be seen that the accuracy results of all investigated methods are generally high. As the data designer indicates, one of the reasons lies in the fact that respondents pay more attention to process behavior than process structure when sorting related processes. The TER method proposed in this work is superior to most of the existing behavior similarity methods in terms of accuracy, and more in line with the results of the previous studies by other scholars.

VII. CONCLUSION AND FUTURE WORK

Behavior differentiation of similar-but-different process variants is an important research subject in business process management. At present, the latest process similarity measurement methods cannot evaluate the similarity accurately by taking process behavior semantics into consideration, nor can they work effectively when invisible tasks are contained in the process structures. To solve the problem, this research proposes an approach of process similarity measurement based on task execution relationship, which takes the behavior profiles of tasks into consideration, and the corresponding similarity evaluation method is called TER method for simplicity. More importantly, the biggest innovation of the TER method lies in the fact that, even under the scenarios that invisible tasks are included in the process structures, the TER matrices evaluating the execution task relationship of all tasks can achieve the expected target of distinguishing dynamic semantics of different behaviors of process variants. In other words, as long as the TER matrices of two process models are different, the behaviors of the corresponding process variants must be different. And it is proved theoretically in this research.

In the experiment part, a series of experiments are designed to evaluate four perspectives of the TER method, including the perspectives of effectiveness, semantic uniqueness expressing, correctness and accuracy. The experimental results show that the proposed method is superior to the current typical process similarity methods in terms of accuracy.

It is undeniable that, while the proposed method in this paper has the above-mentioned advantages, it also has some shortcomings. For example, it has longer execution time than typical process similarity measurement methods, because there exists a procedure dealing with invisible tasks; however, as the proposed method is based on process models and it has the complexity of $O(n^2)$, the total execution time of proposed method is also in an acceptable range. As for future work, we will focus on the performance improvement of the proposed method, specifically when the method is applied to large and complex process models.

Besides, to further the study, we will carry out research into how to retrieve process models based on the existing TER method, and solve the problem of handling non-free choice structures. In addition, it is assumed that the task alignments of two counterpart processes is well defined, so the tag matching issue between task nodes is not taken into consideration in this paper. Therefore, in future studies, the TER method in this paper can be improved by considering duplicated tasks.

ACKNOWLEDGMENT

The authors gratefully acknowledge the helpful comments and suggestions of the reviewers, which have greatly improved the presentation.

REFERENCES

[1] M. Döhring, H. A. Reijers, and S. Smirnov, "Configuration vs. adaptation for business process variant maintenance: An empirical study," *Inf. Syst.*, vol. 39, pp. 108–133, Jan. 2014, doi: 10.1016/j.is.2013.06.002.

[2] W. M. P. van der Aalst, *Process Mining: A 360 Degree Overview*. Cham, Switzerland: Springer, 2022, pp. 3–34.

[3] A. Hmami, H. Sbai, and M. Fredj, "Enhancing change mining from a collection of event logs: Merging and filtering approaches," *J. Physics: Conf. Ser.*, vol. 1743, no. 1, Jan. 2021, Art. no. 012020, doi: 10.1088/1742-6596/1743/1/012020.

[4] A. Khannat, H. Sbai, and L. Kjiri, "Configurable process mining: Semantic variability in event logs," in *Proc. 23rd Int. Conf. Enterprise Inf. Syst. (ICEIS)*, vol. 1, Apr. 2021, pp. 768–775, doi: 10.5220/0010484207680775.

[5] W. M. P. van der Aalst, *Process Mining: Discovery, Conformance and Enhancement of Business Processes*. The Netherlands: Springer, 2011, pp. 12–40.

[6] J. Pei, L. Wen, X. Ye, and A. Kumar, "Efficient transition adjacency relation computation for process model similarity," *IEEE Trans. Services Comput.*, vol. 15, no. 3, pp. 1295–1308, May 2022, doi: 10.1109/TSC.2020.2984605.

[7] A. A. Amer, H. I. Abdalla, and L. Nguyen, "Enhancing recommendation systems performance using highly-effective similarity measures," *Knowl.-Based Syst.*, vol. 217, Apr. 2021, Art. no. 106842, doi: 10.1016/j.knosys.2021.106842.

[8] M. Dumas, G. B. Luciano, and R. M. Dijkman, "Similarity search of business process models," *IEEE Data Eng. Bull.*, vol. 32, pp. 23–28, 2009. [Online]. Available: https://www.researchgate.net/publication/220282702_Similarity_Search_of_Business_Process_Models

[9] A. Schoknecht, T. Thaler, P. Fettke, A. Oberweis, and R. Laue, "Similarity of business process models—A state-of-the-art analysis," *ACM Comput. Surveys*, vol. 50, no. 4, pp. 1–33, Jul. 2018, doi: 10.1145/3092694.

[10] M. Pourbafrani, S. J. van Zelst, and W. M. P. van der Aalst, "Supporting automatic system dynamics model generation for simulation in the context of process mining," in *Proc. Bus. Inf. Syst., 23rd Int. Conf. (BIS)*, in Lecture Notes in Business Information Processing, A. Witold and K. Gary, Eds. Colorado Springs, CO, USA: Springer, vol. 389, Jun. 2020, pp. 249–263, doi: 10.1007/978-3-030-53337-3_19.

[11] M. L. Rosa, W. M. P. V. D. Aalst, M. Dumas, and F. P. Milani, "Business process variability modeling: A survey," *ACM Comput. Surveys*, vol. 50, no. 1, pp. 1–45, Jan. 2018, doi: 10.1145/3041957.

[12] F. Taymouri, M. L. Rosa, M. Dumas, and F. M. Maggi, "Business process variant analysis: Survey and classification," *Knowledge-Based Syst.*, vol. 211, Jan. 2021, Art. no. 106557, doi: 10.1016/j.knosys.2020.106557.

[13] D. M. M. Schunselaar, V. Eric, W. M. P. van der Aalst, and H. A. Raijers, "Creating sound and reversible configurable process models using CoSeNets," in *Proc. Bus. Inf. Syst., 15th Int. Conf. (BIS)*, in Lecture Notes in Business Information Processing, A. Witold, K. Dalia, and S. Virgilijus, Eds. Berlin, Germany: Springer, vol. 117, May 2012, pp. 24–35, doi: 10.1007/978-3-642-30359-3_3.

[14] N. van Beest, H. Groefsema, L. García-Bañuelos, and M. Aiello, "Variability in business processes: Automatically obtaining a generic specification," *Inf. Syst.*, vol. 80, pp. 36–55, Feb. 2019, doi: 10.1016/j.is.2018.09.005.

[15] L. Van Den Ingh, R. Eshuis, and S. Gelper, "Assessing performance of mined business process variants," *Enterprise Inf. Syst.*, vol. 15, no. 5, pp. 676–693, May 2021, doi: 10.1080/17517575.2020.1746405.

[16] H. Fang, W. Liu, W. Wang, and S. Zhang, "Discovery of process variants based on trace context tree," *Connection Sci.*, vol. 35, no. 1, pp. 1–29, Dec. 2023, doi: 10.1080/09540091.2023.2194578.

[17] A. Pourmasoumi, M. Kahani, and E. Bagheri, "Mining variable fragments from process event logs," *Inf. Syst. Frontiers*, vol. 19, no. 6, pp. 1423–1443, Dec. 2017, doi: 10.1007/s10796-016-9662-x.

[18] F. Zandkarimi, J. R. Rehse, P. Soudmand, and H. Hoehle, "A generic framework for trace clustering in process mining," in *Proc. 2nd Int. Conf. Process Mining (ICPM)*, Padua, Italy, B. F. van Dongen, M. Montali, and M. T. Wynn, Eds. Oct. 2020, pp. 177–184, doi: 10.1109/ICPM49681.2020.00034.

[19] Z. Tariq, D. Charles, S. McClean, I. McChesney, and P. Taylor, "An event-level clustering framework for process mining using common sequential rules," in *Proc. Emerg. Technol. Comput., 4th EAI/IAER Int. Conf. (iCETiC)*, M. H. Miraz, G. Southall, M. Ali, A. Ware, and S. Soomro, Eds. Cham, Switzerland: Springer, vol. 395, Aug. 2021, pp. 147–160, doi: 10.1007/978-3-030-90016-8_10.

[20] C. Zhou, Q. Zeng, C. Liu, H. Duan, and G. Yuan, "Business process similarity computing method based on process model structure and log behavior," *Comput. Integr. Manuf. Syst.*, vol. 24, no. 7, pp. 1793–1805, 2018, doi: 10.13196/j.cims.2018.07.021.

[21] A. Dabah, I. Chegrane, and S. Yahiaoui, "Efficient approximate approach for graph edit distance problem," *Pattern Recognit. Lett.*, vol. 151, pp. 310–316, Nov. 2021, doi: 10.1016/j.patrec.2021.08.027.

[22] Z. X. Wang, L. J. Wen, S. H. Wang, and J. M. Wang, "Similarity measurement for process models based on transition-labeled graph edit distance," *Comput. Integr. Manuf. Syst.*, vol. 22, no. 2, pp. 343–352, 2016, doi: 10.13196/j.cims.2016.02.007.

[23] M. Yin, L. Wen, J. Wang, X. Han, Z. Ding, and X. Gao, "Process similarity algorithm based on importance of transition adjacent relations," *Comput. Integr. Manuf. Syst.*, vol. 21, no. 2, pp. 113–129, 2015, doi: 10.13196/j.cims.2015.02.007.

[24] J. Pei, L. Wen, X. Ye, and A. Kumar, "Efficient translation adjacency relation computation for process model similarity," *IEEE Trans. Service Comput.*, vol. 15, pp. 1295–1308, 2022, doi: 10.1109/TSC.2020.2984605.

[25] S. Jinfeng, W. Lijie, and W. Jianmin, "A similarity measure for process models based on task occurrence relations," *J. Comput. Res. Develop.*, vol. 54, no. 4, pp. 832–843, 2017, doi: 10.7544/issn1000-1239.2017.20151176.

[26] L. Wen, J. Song, J. Wang, and A. Kumar. *BP^+: An Improved Behavioral Profile Metric for Process Models*. Accessed: Dec. 2015. [Online]. Available: http://www.researchgate.net/publication/286932844

[27] M. Weidlich, J. Mendling, and M. Weske, "Efficient consistency measurement based on behavioral profiles of process models," *IEEE Trans. Softw. Eng.*, vol. 37, no. 3, pp. 410–429, May 2011, doi: 10.1109/TSE.2010.96.

[28] M. Weidlich, A. Polyvyanyy, J. Mendling, and M. Weske, "Causal behavioural profiles–efficient computation, applications, and evaluation," *Fundamenta Informaticae*, vol. 113, nos. 3–4, pp. 399–435, 2011, doi: 10.3233/FI-2011-614.

[29] R. Duan, H. Fang, and Y. Zhan, "Process similarity algorithm based on weighted flow relationship," *Acta Electonica Sinica*, vol. 47, no. 12, pp. 2596–2601, 2019. [Online]. Available: https://www.ejournal.org.cn/CN/Y2019/V47/I12/2596

[30] D. Li and H. Fang, "An approach of process similarity measurement based on activity occurrence relationship," *Control Theory Appl.*, vol. 37, no. 9, pp. 2011–2019, 2020. [Online]. Available: http://jcta.alljournals.ac.cn/cta_cn/ch/reader/view_abstract.aspx?file_no=CCTA190768&flag=1

[31] C. Zhou, C. Liu, Q. Zeng, Z. Lin, and H. Duan, "A comprehensive process similarity measure based on models and logs," *IEEE Access*, vol. 7, pp. 69257–69273, 2019, doi: 10.1109/ACCESS.2018.2885819.

[32] C. Liu, Q. Zeng, L. Cheng, H. Duan, and J. Cheng, "Measuring similarity for data-aware business processes," *IEEE Trans. Autom. Sci. Eng.*, vol. 19, no. 2, pp. 1070–1082, Apr. 2022, doi: 10.1109/TASE.2021.3049772.

[33] C. Liu, Q. Zeng, H. Duan, S. Gao, and C. Zhou, "Towards comprehensive support for business process behavior similarity measure," *IEICE Trans. Inf. Syst.*, vol. E102.D, no. 3, pp. 588–597, 2019, doi: 10.1587/transinf.2018EDP7127.

[34] L. Wang, Y. Du, and L. Qi, "Efficient deviation detection between a process model and event logs," *IEEE/CAA J. Autom. Sinica*, vol. 6, no. 6, pp. 1352–1364, Nov. 2019, doi: 10.1109/JAS.2019.1911750.

[35] Z. Nagy and A. Werner-Stark, "An alignment-based multi-perspective online conformance checking technique," *Acta Polytechnica Hungarica*, vol. 19, no. 4, pp. 105–127, 2022. [Online]. Available: https://acta.uni-obuda.hu/Nagy_Werner-Stark_122.pdf

[36] A. Tsoury, P. Soffer, and I. Reinhartz-Berger, "How well did it recover? Impact-aware conformance checking," *Computing*, vol. 103, pp. 3–27, Nov. 2021, doi: 10.1007/s00607-020-00857-y.

[37] S. Zhang, L. Genga, L. Dekker, H. Nie, X. Lu, H. Duan, and U. Kaymak, "Fuzzy multi-perspective conformance checking for business processes," *Appl. Soft Comput.*, vol. 130, Nov. 2022, Art. no. 109710, doi: 10.1016/j.asoc.2022.109710.

[38] P. H. P. Richetti, L. S. Jazbik, F. A. Baião, and M. L. M. Campos, "Deviance mining with treatment learning and declare-based encoding of event logs," *Exp. Syst. Appl.*, vol. 187, Jan. 2022, Art. no. 115962, doi: 10.1016/j.eswa.2021.115962.

[39] M. Alizadeh, X. Lu, D. Fahland, N. Zannone, and W. M. P. van der Aalst, "Linking data and process perspectives for conformance analysis," *Comput. Secur.*, vol. 73, pp. 172–193, Mar. 2018, doi: 10.1016/j.cose.2017.10.010.

[40] S. J. J. Leemans and A. Polyvyanyy, "Stochastic-aware precision and recall measures for conformance checking in process mining," *Inf. Syst.*, vol. 115, May 2023, Art. no. 102197, doi: 10.1016/j.is.2023.102197.

[41] L. Cheng, C. Liu, and Q. Zeng, "Optimal alignments between large event logs and process models over distributed systems: An approach based on Petri nets," *Inf. Sci.*, vol. 619, pp. 406–420, Jan. 2023, doi: 10.1016/j.ins.2022.11.052.

[42] L. Genga, M. Alizadeh, D. Potena, C. Diamantini, and N. Zannone, "Discovering anomalous frequent patterns from partially ordered event logs," *J. Intell. Inf. Syst.*, vol. 51, no. 2, pp. 257–300, Oct. 2018, doi: 10.1007/s10844-018-0501-z.

[43] T. Murata, "Petri nets: Properties, analysis and applications," *Proc. IEEE*, vol. 77, no. 4, pp. 541–580, Apr. 1989, doi: 10.1109/5.24143.

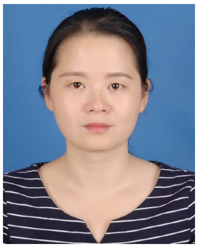
[44] Z. Ma, Z. He, Z. Li, and A. Giua, "Design of supervisors for linear marking specifications in labeled Petri nets," *Automatica*, vol. 136, Feb. 2022, Art. no. 110031, doi: 10.1016/j.automatica.2021.110031.

[45] G. Liu, *Petri Nets: Theoretical Models and Analysis Methods for Concurrent Systems*, 1st ed. Cham, Switzerland: Springer, 2022, ch. 1, sec. 2, pp. 6–12.

[46] L. Wang and X. Fang, "Mining approach of process model with hidden transition based on the behavioral distance," *Comput. Integr. Manuf. Syst.*, vol. 27, no. 9, pp. 2670–2679, 2021, doi: 10.13196/j.cims.2021.09.019.

[47] B. Cao, J. Wang, W. An, J. Fan, and S. Chen, "ProBench: A benchmark dataset for evaluating the process similarity search methods," *Comput. Integr. Manuf. Syst.*, vol. 23, no. 5, pp. 1069–1079, 2017, doi: 10.13196/j.cims.2017.05.018.

[48] Y. Wang, L. Wang, Y. Li, D. He, W. Chen, and T. Liu, "A theoretical analysis of NDCG ranking measures," in *Proc. 26th Annu. Conf. Learn. Theory (COLT)*. Princeton, NJ, USA: Princeton University, Jun. 2013, pp. 1–2, doi: 10.48550/arXiv.1304.6480.



JUAN GUO received the B.S. degree in computer science and engineering from the Shandong University of Science and Technology, in 2002, and the M.S. degree in computer science and engineering from the Anhui University of Science and Technology, in 2009.

Since 2009, she has been a Lecturer with the School of Mathematics and Big Data, Anhui University of Science and Technology. Her research interests include petri nets, behavior profiles, and process mining.

HUAN FANG received the B.S. degree in information and computing from the Anhui University of Science and Technology, Huainan, China, in 2003, the M.S. degree in computer science and engineering from the Shandong University of Science and Technology, China, in 2006, and the Ph.D. degree in computer science and engineering from the Hefei University of Technology, China, in 2013.

From 2018 to 2021, she was a Postdoctoral Researcher with the Anhui University of Science and Technology, where she is currently a Professor with the School of Mathematics and Big Data. Her research interests include petri nets, process mining, change mining, and intelligent control.

Prof. Fang's awards and honors includes the Excellent Teachers of Anhui Province, China.