Hindawi
Scientific Programming
Volume 2022, Article ID 6993449, 13 pages
https://doi.org/10.1155/2022/6993449

Research Article

Deviation Detection in Clinical Pathways Based on Business Alignment

Yinhua Tian,[1] **Xinran Li**,[1] **Man Qi**,[2] **Dong Han**,[3] **and Yuyue Du**[4]

[1]*College of Intelligent Equipment, Shandong University of Science and Technology, Tai'an, Shandong 271000, China*
[2]*School of Engineering, Technology and Design, Canterbury Christ Church University, Canterbury CT1 1QU, UK*
[3]*College of Continuing Education, Shandong University of Science and Technology, Tai'an, Shandong 271000, China*
[4]*College of Computer Science and Engineering, Shandong University of Science and Technology, Qingdao, Shandong 266590, China*

Correspondence should be addressed to Dong Han; aa1130_2011@163.com

Received 19 November 2021; Revised 8 December 2021; Accepted 14 December 2021; Published 6 January 2022

Academic Editor: Tongguang Ni

Several unexpected behaviors may occur during actual treatment of clinical pathways, which will have negative impact on the implementation and the future work. To increase the performance of current deviation detection algorithms, a method is presented according to business alignment, which can effectively detect the anomaly in the implementation of the clinical pathways, provide judgment basis for the intervention in the process of the clinical pathway implementation, and play a crucial role in improving the clinical pathways. Firstly, the noise in diagnosis and treatment logs of clinical pathways will be removed. Then, the synchronous composition model is constructed to embody the deviations between the actual process and the theoretical model. Finally, A $*$ algorithm is selected to search for optimal alignment. A clinical pathway for ST-Elevation Myocardial Infarction (STEMI) under COVID-19 is used as a case study, and the superiority and effectiveness of this method in deviation detection are illustrated in the result of experiments.

1. Introduction

The data released by the seventh census show that China has gradually entered an aging society. Coupled with the impact of the coronavirus epidemic in recent years, the medical system is facing increasing pressure. Hence, the use of clinical pathway that can regulate medical behavior, improve medical quality, and control medical costs has become an inevitable choice for medical reform in China [1, 2]. With strong national support, the clinical pathway has entered the stage of large-scale promotion [3].

The essence of clinical pathway is to adopt standardized procedures for the diagnosis and treatment of a certain disease. However, in the actual implementation process of clinical pathway, there may be many unstable factors; for instance, patients do not understand the implementation standard of clinical pathway, medical staff are not positive about the process, etc. Factors mentioned above have greatly affected the promotion of clinical pathway. In addition, the imperfection of clinical pathway, including patient condition mutation, may lead to unexpected behaviors in the process of clinical pathway diagnosis and treatment. Such behaviors are all called deviations. Under various pressures, it is urgent to optimize and improve the clinical pathway. Process mining aims to mine and optimize the business process through effective data in event logs [4, 5], and the process mining methods can be used to detect the deviated behaviors, which is beneficial to predict the trend of the patient's diagnosis and treatment process, find the defects in the clinical pathway, and provide judgment basis for the intervention behavior in the diagnosis and treatment process, so as to improve the clinical pathway [6–8].

With the wide application of medical information systems, a large number of electronic logs are generated in the process of diagnosis and treatment, and electronic medical records are also widely used in hospitals. The method of

mining deviation behaviors has also changed from a forward-looking method requiring manual recording to a retrospective method automatically processed by computers. This paper establishes the process model of clinical pathway and uses business alignment to detect the deviation between the business process and the execution records.

Lots of formal methods of process model description have been presented up to now, e.g., BPMN [9], C-net [10], EPC [11] and Petri net [12–15]. Petri net is the most mature and in-depth process modeling language, which can concisely and intuitively describe and analyze complex systems [16–27]. Hence, Petri net is selected as the process model description method.

Business alignment is one of the most advanced compliance checking methods at present. The use of business alignment can effectively detect the deviation of clinical pathway [28–30]. Compliance checking of complex models has always been a challenging research topic [31]. Adriansyah et al. [32] presented a technique which combines the process model and log model into a product model to get optimal alignment. Cook et al. [33] presented a means to obtain alignment results by comparing trace and process model with quantifying similarity. Song et al. [34] presented a heuristic trace replay method to align recorded traces and theoretical models, which reduces the search space.

By studying the existing methods, there is still something that can be improved in the efficiency of business alignment. Hence, a business alignment method is presented according to synchronous composition model of Petri nets, which effectively reduces the complexity of the model to improve alignment efficiency. A sequence of events recorded in the medical log of a case is called a trace. There may be some invalid traces in the log, which are called noise. Before the compliance check, it is necessary to filter the diagnosis and treatment logs by preprocessing to remove noise. Deviation detection is to find the optimal alignment based on the filtered treatment logs and locate possible problems in the traces. Since the efficiency of the optimal alignment computation is particularly critical, this paper uses A$*$ algorithm.

The remainder of this paper includes the following. The background knowledge is introduced in Section 2, including the concept of trace, Petri net, reachable graph, and alignment. A noise filter algorithm and an optimal alignment computation algorithm are presented in Section 3. The comparison of the experiments between our method and the classical one is shown in Section 4. Finally, Section 5 draws the conclusion and the ideas of follow-up research.

2. Basic Knowledge

A tremendous amount of data is stored in hospital information systems. We can extract a lot of medical events from them. A series of events in one case is organized as a trace, which is the basic element of logs.

Definition 1 (trace). Given a set A that includes activities, trace is a process instance, namely, a sequence of activities, denoted as $\sigma \in A^*$. If there is a trace of non-empty multiple set $L \in \beta(A^*)$, L is called a diagnosis and treatment log, and the set includes each finite sequence on set A, denoted as A^*; the set of all multiple sets on set A^* is denoted as $\beta(A^*)$.

Definition 2 (Petri net). Given a set A that includes activities, Petri net is a tuple on set A, denoted as $N = (P, T; F, \alpha, m_i, m_f)$. Set P contains all places, and set T contains all transitions, where $T \cup P \neq \varnothing$, $T \cap P = \varnothing$; set $F \subseteq (T \times P) \cup (P \times T)$ represents the directed arc set of the relationship between the place and the transition, which is called the flow relationship or arc relationship; $\alpha: T \longrightarrow A^\tau$ is the mapping function of transition and label, and τ is the invisible transition, that is, $A^\tau = A \bigcup \{\tau\}$; the state of Petri net is named as marking, which is a multiple set of the place set. $m_i \in \beta(P)$ stands for the initial state, and $m_f \in \beta(P)$ stands for the final state.

A simple Petri net example N_1 for clinical pathway of ischemic stroke is shown in Figure 1, and Table 1 shows the relationship of transitions, labels, and activities in model N_1.

For $\forall m \in \beta(P)$ and $\forall t \in T$, t can be fired under marking m iff $m(p) \geq \forall_{p \in P} F(p, t)$ holds, represented as $m[t>$; at this time, after the transition t occurs, the new state of the system denotes as m', and $\forall_p \in {}_P m'(p) = m(p) - F(p, t) + F(t, p)$, represented as $m[t > m'$. The set $R(m)$ contains all the markings which can be enabled under m.

The definition of Petri net's reachable graph can be constructed by the transition firing rule in Definition 2.

Definition 3 (reachable graph). Given a Petri net N, reachable graph of N is represented as $TS = (S, A', T')$, where $S = R(m_i)$, $A' = A$, $T' = \{(m_i, \alpha(t), m_j) \in (S \times A \times S) \mid \exists_{t \in T} (m_i[t > m_j])\}$.

Figure 2 shows the reachable graph TS_0 of N_1. Every node in the graph stands for a reachable marking of N_1, every arc means a transition that is enabled under its previous marking, and arrow points to the new marking after transition occurs.

When the trace in the diagnosis and treatment log is replayed on the model, the two may not be completely fitted. This unfit state is called deviation which can be detected by alignment.

Definition 4 (alignment) Given a set A including activities, a Petri net N, and a trace σ, the alignment between σ and A, namely, $\gamma \in (A\gg \times T\gg)^*$ is a moving sequence (where $\gg$ denotes no movement and $A\gg = A \cup \{\gg\}$) which has the following rules:

(1) The projection of the first column elements in the ordered set of movements on A (ignoring $\gg$) is recorded as $\pi_1(\gamma)_{\downarrow A} = \sigma$.

(2) The projection of the second column elements in the ordered set of movements on T (ignoring $\gg$) produces a complete firing sequence, denoted as $m_i \longrightarrow^{\pi_2} (\gamma)_{\downarrow T} m_f$.

For every $(a, t) \in \gamma$, there are four possibilities:

(1) (a, t) is named as log movement in case of $(a \in A) \wedge (t = \gg)$.

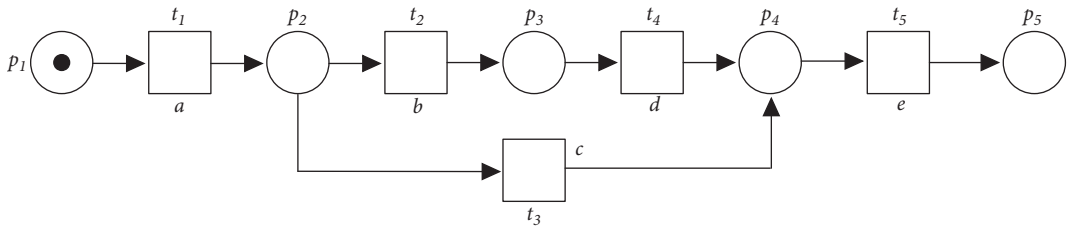

Figure 1: Petri net N_1 for clinical pathway of ischemic stroke.

Table 1: The relationship between transitions and activities of model N_1.

Transition	Label	Activity	Transition	Label	Activity
t_1	a	Initial inspection	t_4	d	Thrombolytic therapy
t_2	b	Meeting thrombolytic inclusion criteria	t_5	e	Hospitalization
t_3	c	Meeting thrombolytic exclusion criteria			

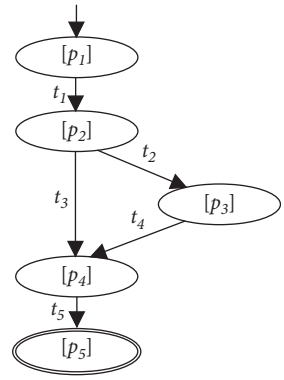

Figure 2: Reachable graph TS_0 for N_1.

(2) (a, t) is named as model movement in case of $(a = \gg) \wedge (t \in T)$.

(3) (a, t) is named as synchronous movement in case of $(a \in A) \wedge (t \in T)$.

(4) (a, t) is named as illegal movement.

Activity is an element of traces, and transition is an element of model. According to Definition 4, alignment is a sequence of movements, and movement reflects the correlation between the activities and transitions. For log movement, it means that an activity cannot be executed in the model; for model movement, it means that a transition is not observed in the trace; for synchronous movement, it means that the activity can correspond to the transition; for illegal movement, it will not occur in actual business process, so it will be ignored in this paper. Among them, (1) and (2) are the unfitness between trace and process model, which represent the deviations in alignment.

For a given trace and a process model, multiple alignment results may be computed. To get the best alignment result, a cost function needs to be introduced to compute the cost of each movement. Among all the alignment results, the alignment with the minimum total cost value is the optimal alignment.

For every $(a, t) \in \gamma$, the cost function is as follows:

(1) The value of $lc((a, t))$ is 1, when (a, t) is log movement.

(2) The value of $lc((a, t))$ is also 1, when (a, t) is model movement.

(3) The value of $lc((a, t))$ is 0, when (a, t) is synchronous movement.

(4) The value of $lc((a, t))$ is $+\infty$, when (a, t) is illegal movement.

3. Alignment Methods

3.1. Preparation for Alignment Computation

3.1.1. Removal of Noise. Before the implementation of synchronous composition, it is necessary to filter the traces in the diagnosis and treatment log, remove the noise, and retain effective traces.

There are generally two possibilities for traces with sequential deviation in event logs: ① the trace itself is noise and ② the activity which should occur in sequential relationship or selection relationship occurred in parallel relationship incorrectly. In the clinical pathway, it must be strictly in accordance with the implementation standards of clinical pathway, and the activities are mostly sequential or selective. Hence, traces with inverted sequence activities are considered as noise in this paper.

In Petri net, when a transition can occur before other transitions, there is a precedence relationship between this transition and its subsequent transitions, which is denoted by the symbol "⟶." When a transition cannot occur before others, there is no precedence relationship between this transition and others, which is denoted by the symbol "#." For the possible sequential deviation traces in the diagnosis and treatment log, this paper presents the concept of precedence relationship matrix as follows.

Definition 5. (precedence relationship matrix). Given a set A including activities and a Petri net N, the precedence relationship matrix is a matrix of $|T| \times |T|$, whose row and column labels are the activities, denoted as $M{:}\{\alpha(t_x)|t_x \in T\} \times \{\alpha(t_y)|t_y \in T\} \longrightarrow \{\text{"\#," "}\longrightarrow\text{"}\}$. The matrix contains the following elements:

(1) For any $\alpha(t_x) \in S$ and $\alpha(t_y) \in S$, if $\exists < t_1, \ldots, t_x, \ldots, t_y, \ldots, t_n >$, $m_i \longrightarrow^{t_1} \ldots t_x \ldots t_y \ldots t_n\, m_f$, then $M[\alpha(t_x)][\alpha(t_y)] =$ "⟶," denoted as $\alpha(t_x) \longrightarrow \alpha(t_y)$, where $1 \le x, y \le n$.

(2) For any $\alpha(t_x) \in S$ and $\alpha(t_y) \in S$, if $\nexists < t_1, \ldots, t_x, \ldots, t_y, \ldots, t_n >$, $m_i \longrightarrow^{t_1} \ldots t_x \ldots t_y \ldots t_n\, m_f$, then $M[\alpha(t_x)][\alpha(t_y)] =$ "#," denoted as $\alpha(t_x) \# \alpha(t_y)$, where $1 \le x, y \le n$.

Take N_1 as an example, and its precedence relationship matrix M_{pr} is shown in formula (1), where $a_x = \alpha(t_x)$ and $1 \le x \le 5$.

$$M_{pr} = \begin{array}{c} \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{array} \begin{array}{c} \begin{array}{ccccc} a_1 & a_2 & a_3 & a_4 & a_5 \end{array} \\ \begin{bmatrix} \# & \longrightarrow & \longrightarrow & \longrightarrow & \longrightarrow \\ \# & \# & \# & \longrightarrow & \longrightarrow \\ \# & \# & \# & \# & \longrightarrow \\ \# & \# & \# & \# & \longrightarrow \\ \# & \# & \# & \# & \# \end{bmatrix} \end{array}. \quad (1)$$

To determine whether a trace is noise, it is necessary to check all activities in the trace one by one and compare the order between the current activity and its subsequent activities by the precedence relationship matrix. If not fitting, the comparison process is terminated, and the trace is regarded as noise, which should be removed from the log.

To store information of parallel activities, it is necessary to set a parallel activity set S as "current activity set" to compare with their subsequent activities, and the first activity in the trace is put into S before starting the traversal. Meanwhile, several variables are set as follows: let the current activity in traversal be activity *cur* and the subsequent activity in trace be activity *post*. The rules are as follows:

(1) If there is one or more activities that precedence relationship matrix allow to occur earlier than activity *post* in the set S and otherwise do not hold, remove these activities from the set S and place activity *post* in the set S.

(2) If all activities in the set constitute the parallel relationship with activity *post*, then put activity *post* in the set S without removing any other element.

(3) If there is an inverted sequence deviation between any activity and activity *post* in set S, this trace is regarded as noise and should be abandoned directly.

Algorithm 1 shows the pseudocode.

To filter the log, it is necessary to traverse the diagnosis and treatment log L and then gradually check the activities in the sequence σ, so two nested loops are required. $O(n^2)$ is its time complexity. The value of n is affected by the size of L and the length of the sequence σ. The space complexity is $O(n)$.

3.1.2. Synchronous Composition. For a given trace and its corresponding process model, a new model can be generated by the following operations: ① convert the trace into the log model described by Petri net; ② merge two corresponding transitions with the same activity label in two models; and ③ merge presets and postsets of the same transitions, respectively. Finally, the generated model is the synchronous composition model of trace and process model.

The log model transition and process model transition with the same label x will construct synchronous movement. For other synchronous movements, the transitions should be named according to Definition 4. Algorithm 2 shows the pseudocode.

The algorithm integrates the transitions, places, and arc relationships of process model N_{pm} and log model N_{lm} into synchronous composition model N_3. There are three loop structures, so $O(n)$ is the time complexity. The quantity of transitions, places, and the arc relationships affect the size of n, so space complexity is $O(n)$.

Taking N_1 as an example, trace $\sigma_1 = <a, f, b, e>$ is given. Figure 3 shows the log model N_2 which is converted from σ_1. The synchronous composition model N_3 computed by Algorithm 2 from log model N_2 and process model N_1 is shown in Figure 4.

3.2. Computation of Alignment

3.2.1. Reachable Graph of Synchronous Composition Model. Since alignment is a sequence of movements, the transitions occur with the change of states and reachable graph can clearly express the changes of states and occurred transitions, and a weight can be given to every arc of a reachable graph to represent the cost of the occurred transition. The computation of optimal alignment can be converted to find the shortest path of a directed weighted graph, so the reachable graph of N_3 is then computed. The reachable graph TS_1 of N_3 is shown in Figure 5.

It can be inferred from Definition 4 that (t_1', t_1) is a synchronous movement, so $lc((t_1', t_1)) = 0$; $(\gg, t_3)$ is a model movement, so $lc((\gg, t_3)) = 1$; $(t_2', \gg)$ is a log movement, so $lc((t_2', \gg)) = 1$. By analogy, it can be obtained that the values of the other transitions in Figure 5 are $lc((t_3', t_2)) = 0$, $lc((\gg, t_4)) = 1$, $lc((t_4', t_5)) = 0$.

Input: diagnosis and treatment log L, precedence relationship matrix M of N_1;
Output: filtered log L'.

(1) $S = \emptyset$;
(2) $L' = L$;
(3) **for all** $\sigma \in L'$ **do**
(4) $S = S \cup \{\sigma\ (1)\}$;
(5) **for all** $cur \in \sigma$ **do**
(6) //terminate if current activity is the last one
(7) **if** $\sigma.indexOf\ (cur) == \sigma.size - 1$ **then**
(8) **break**;
(9) **end**
(10) $post = \sigma\ (\sigma.indexOf\ (cur) + 1)$;
(11) **if** $(\exists a \in S) \Rightarrow (M\ [\alpha^{-1}\ (a)][\alpha^{-1}(post)] == \text{“\#”})$ **then**
(12) $L' = L' - \{\sigma\}$;
(13) //activity a can occur earlier than activity $post$, but the reverse is not true
(14) **else if** $(\exists a \in S) \Rightarrow (M\ [\alpha^{-1}\ (a)][\alpha^{-1}(post)] == \text{“}\longrightarrow\text{”}\ \&\&\ M\ [\alpha^{-1}(post)][\alpha^{-1}(a)] == \text{“\#”})$ **then**
(15) **for all** $(a \in S) \wedge (M\ [\alpha^{-1}(a)][\alpha^{-1}(post)] == \text{“}\longrightarrow\text{”}\ \&\&\ M\ [\alpha^{-1}(post)][\alpha^{-1}(a)] == \text{“\#”})$ **do**
(16) $S = S - \{a\}$;
(17) **end**
(18) $S = S \cup \{post\}$;
(19) **else**
(20) $S = S \cup \{post\}$;
(21) **end**
(22) **end**
(23) $S = \emptyset$;
(24) **end**
(25) **return** L';

Algorithm 1: The noise filter algorithm.

Input: process model N_{pm}, log model N_{lm};
Output: synchronous composition model N_{cm}.

(1) $P_{\text{cm}} = P_{\text{pm}} \cup P_{\text{lm}}$;
(2) $F_{\text{cm}} = \emptyset$;
(3) $T_{\text{cm}} = \emptyset$;
(4) $m_{i,\text{cm}} = m_{i,\text{pm}} \cup m_{i,\text{lm}}$;
(5) $m_{f,\text{cm}} = m_{f,\text{pm}} \cup m_{f,\text{lm}}$;
(6) **for all** $t'_x \in T_{\text{lm}}$ **do**
(7) //place transitions in T_{cm} according to T_{lm};
(8) $T_{\text{cm}} = T_{\text{cm}} \cup \{(t'_x, \gg)\}$;
(9) //set the related mapping functions and arc relations
(10) $\alpha_{\text{cm}}\ ((\gg, t_y)) = \alpha_{\text{pm}}\ (t_y)$;
(11) $F_{\text{cm}} = F_{\text{cm}} \cup \{(p, (\gg, t_y)) | p \in {}^\bullet t_y \wedge t_y \in T_{\text{pm}}\} \cup \{((\gg, t_y), p) | p \in t_y^\bullet \wedge t_y \in T_{\text{pm}}\}$;
(12) **end**
(13) **for all** $t_y \in T_{\text{pm}}$ **do**
(14) //place transitions in T_{cm} according to T_{pm};
(15) $T_{\text{cm}} = T_{\text{cm}} \cup \{(\gg, t_y)\}$;
(16) //set the related mapping functions and arc relationships
(17) $\alpha_{\text{cm}}\ ((\gg, t_y)) = \alpha_{\text{pm}}\ (t_y)$;
(18) $F_{\text{cm}} = F_{\text{cm}} \cup \{(p, (\gg, t_y)) \mid p \in {}^\bullet t_y \wedge t_y \in T_{\text{pm}}\} \cup \{((\gg, t_y), p) \mid p \in t_y^\bullet \wedge t_y \in T_{\text{pm}}\}$;
(19) **end**
(20) **for all** $t'_x \in T_{\text{lm}} \wedge t_y \in T_{\text{pm}} \wedge \alpha_{\text{lm}}\ (t'_x) == \alpha_{\text{pm}}\ (t_y)$ **do**
(21) //place synchronous transitions in T_{cm};
(22) $T_{\text{cm}} = T_{\text{cm}} \cup \{(t'_x, t_y)\}$;
(23) //remove log transitions and model transitions with the same labels;
(24) $\alpha_{\text{cm}}\ ((t'_x, t_y)) = \alpha_{\text{cm}}\ (t'_x, \gg)$;
(25) $T_{\text{cm}} = T_{\text{cm}} - \{(t'_x, \gg), (\gg, t_y)\}$;

Algorithm 2: Continued.

(26) //set the related mapping functions and arc relationships of the new transitions; remove the related arc relationships of the removed transitions of the deleted transitions
(27) $F_{cm} = F_{cm} \cup \{((t_x', t_y), p') | p' \in (t_x', \gg)^\bullet\} \cup \{(p', (t_x', t_y)) | p' \in {}^\bullet(t_x', \gg)\}$;
(28) $F_{cm} = F_{cm} - \{((t_x', \gg), p') | p' \in (t_x', \gg)^\bullet\} - \{(p', (t_x', \gg)) | p' \in {}^\bullet(t_x', \gg)\}$;
(29) $F_{cm} = F_{cm} \cup \{((t_x', t_y), p) | p \in (\gg, t_y)^\bullet\} \cup \{(p, (t_x', t_y)) | p \in {}^\bullet(\gg, t_y)\}$;
(30) $F_{cm} = F_{cm} - \{((\gg, t_y), p) | p \in (\gg, t_y)^\bullet\} - \{(p, (\gg, t_y)) | p \in {}^\bullet(\gg, t_y)\}$;
(31) **end**
(32) **return** $N_{cm} = (P_{cm}, T_{cm}; F_{cm}, \alpha_{cm}, m_{i,cm}, m_{f,cm})$;

Algorithm 2: Synchronous composition algorithm.

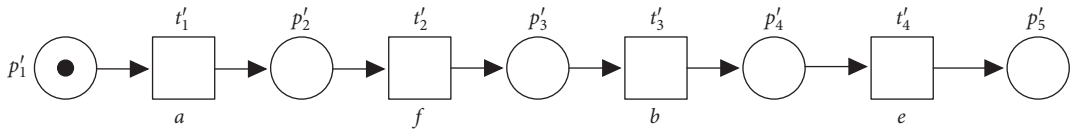

Figure 3: Log model N_2.

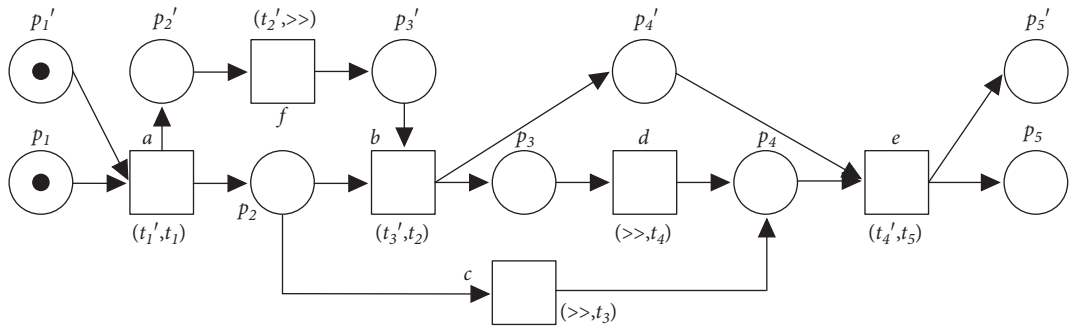

Figure 4: Synchronous composition model N_3.

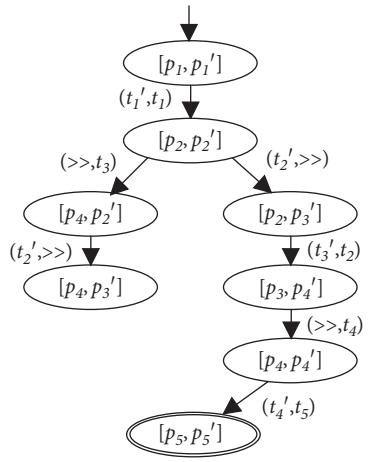

Figure 5: Reachable graph TS_1 of N_3.

3.2.2. Search for Optimal Alignment. Considering the particularity of the clinical pathway model, the sequential and selective structure should be used as far as possible in this kind of model and the cyclic structure should be avoided. Hence, the cyclic structure in the reachable graph of the model will be ignored, and the graph can be easily converted into the form of relationship matrix. It can be seen that the optimal alignment computation is not complicated, and the efficiency is one of the key performance measures. Hence, it is necessary to select appropriate search algorithms to maximize efficiency.

As mentioned earlier, the optimal alignment problem is also the shortest path problem. Among the solutions to the

shortest path problem, the basic search algorithms (e.g., the depth-first algorithm, the breadth-first algorithm, and the Dijkstra algorithm) are well known with simple structure and easy implementation. However, in dealing with this problem, it is necessary to traverse paths one by one, and the efficiency is poor. The intelligent algorithms (e.g., genetic algorithm, reinforcement learning algorithm, and ant colony algorithm) are too complex and need to be trained through a large number of datasets to adjust parameters, which is suitable for solving the tricky issues in complex scenes and is not suitable for solving this problem.

Since the weight of each arc is known, A $*$ algorithm is suitable for this problem, which is a famous heuristic algorithm. When solving the shortest path, A $*$ algorithm will estimate the distance between the current node and the target node. The accuracy of the heuristic function will affect the efficiency of A $*$ algorithm, and the accuracy represents the proximity between the estimated and the actual value. In addition, the code implementation of A $*$ algorithm is simple. Since the estimated value computed by the cost function used in this paper is basically equal to the actual value, the computational efficiency can reach the highest level. For reachable graph TS_1 obtained in Section 3.2.1, Algorithm 3 shows the pseudocode.

Considering that the algorithm uses the priority queue, assuming that it uses quick sort, $O(n\log_2{}^n)$ is the time complexity, and the quantity of reachable markings affects the size of n. In Algorithm 3, the frequency of iterations is affected by the complexity of the reachable graph, too. Hence, the time complexity is $O(n^2\log_2{}^n)$, and the space complexity is $O(n)$.

Taking reachable graph TS_1 as an example, formula (2) is the result of optimal alignment.

$$\gamma_1 = \begin{array}{|c|c|c|c|c|} \hline \text{a} & \text{f} & \text{b} & >> & \text{e} \\ \hline \text{a} & >> & \text{b} & \text{d} & \text{e} \\ \hline t_1 & & t_2 & t_4 & t_5 \\ \hline \end{array} \quad (2) \qquad (2)$$

Algorithm 3 finds the optimal alignment by calculating the minimum sum cost value of the occurred transition sequences. In the implementation of the algorithm, it is not necessary to generate reachable graph of synchronous composition model. The synchronous composition model can be directly used as the input to generate the search space, which can simplify the solving step, save the processing time, and improve the computational efficiency.

4. Experiment Analysis

4.1. Experiment Settings. This experiment will compare the scale of the model and the efficiency of alignment. There are three main elements to measure the scale of the model, including

(1) Quantity of places: how many places there are in the model. In general, the less the number, the lower the complexity.

(2) Quantity of transitions: how many transitions there are in the model. In general, the less the number, the lower the complexity.

(3) Quantity of arcs: how many arc relationships there are in the model. In general, the less the number, the lower the complexity.

The factors that measure alignment efficiency are the time consumed and the memory space occupied by the process of optimal alignment computation. In this experiment, the memory occupancy is measured by the quantity of reachable markings generated in the process of running Petri net. Hence, there are two main factors to measure the alignment algorithm:

(1) Quantity of reachable markings (quantity of queued nodes): how many reachable markings generated there are when the model is running. When calculating the optimal alignment, the number is equal to the quantity of the nodes queued in the priority queue, which is also called as the quantity of queued nodes.

(2) Computation time: the time consumed when computing the optimal alignment.

For a given process model, a log set, and a cost function, the computation process of the optimal alignment which presented by Adriansyah et al. can be divided into two steps: first, generate the search space; then, search for the optimal alignment. The reachable graph of the new-generated model is considered as the search space, whose complexity can be measured by the scale of the new model and basically determine the cost of the entire optimal alignment computation method. This experiment focuses on this part. Similarly, the method in this paper uses A $*$ algorithm for optimal alignment computation. Theoretically, this method reduces unnecessary transitions and arcs, so the quantity of reachable markings is less, and the efficiency should be better than the classical one when computing the optimal alignment.

Next, the clinical pathway of ST-Elevation Myocardial Infarction (STEMI) under COVID-19 is taken as an example for experimental verification. The model is manually established according to the natural language description of the diagnosis and treatment process. The specific clinical pathway is shown in Figure 6, and the relationship between activities and transitions is shown in Table 2.

The main work of this experiment is to compare the scale of the search space and efficiency of the alignment computation between the classical method and the method presented in this paper. All the fitting traces in logs are generated from the running process model, then noise is added in the proportion of the deviation number to the length of the trace from 0% to 30%, and each 5% is a group. Each trace is dealt with by Algorithm 3 and the classical algorithm for 10 times, respectively. The mean computation time and model scale parameters are obtained, so as to fully compare the differences between the two under the premise of control variables.

4.2. Experimental Environment. The experiment code adopts Java language, and Table 3 shows the hardware and software platform configuration.

4.3. Algorithm Superiority Verification. The results are shown in Figures 7–10. Among them, Figures 7 and 8 show the

Input: reachable graph TS of synchronous composition model N_3;
Output: optimal alignment γ.

```
(1)  //initialize a priority queue by (total cost value of m_i to m_f) + (total estimated value of the current node to the target node) in ascending order
(2)  pqueue.create ();
(3)  visitedNodesSet = Ø;
(4)  pqueue.push (TS.initialmarking);
(5)  while pqueue.size ()! = 0 do
(6)    currenrNode = pqueue.poll ();
(7)    if currenrNode == targetNode then
(8)    //recursively search the predecessor node of currenrNode to get optimal alignment
(9)    γ = getOptAlignment (currenrNode);
(10)   return γ;
(11)   else
(12)     //visit all successorNodes of currentNode
(13)     for all successorNode ∈ currentNode.getSuccessors() do
(14)       //calculate the new total cost value of successorNode
(15)       newcost = successorNode.calNewCost (currentNode);
(16)       if successorNode ∈ visitedNodesSet then
(17)         //for the visited node, if the new total cost value is smaller, update the total cost value of successorNode
(18)         if successorNode.getTotalCost() > newcost then
(19)           successorNode .setTotalCost (newcost);
(20)           pqueue.push (successorNode);
(21)         end
(22)       Else
(23)         visitedNodesSet = visitedNodesSet ∪ {successorNode};
(24)         successorNode.setTotalCost (newcost);
(25)         pqueue.push (successorNode);
(26)       end
(27)     end
(28)   end
(29) end
```

Algorithm 3: Optimal alignment algorithm based on A ∗ algorithm.

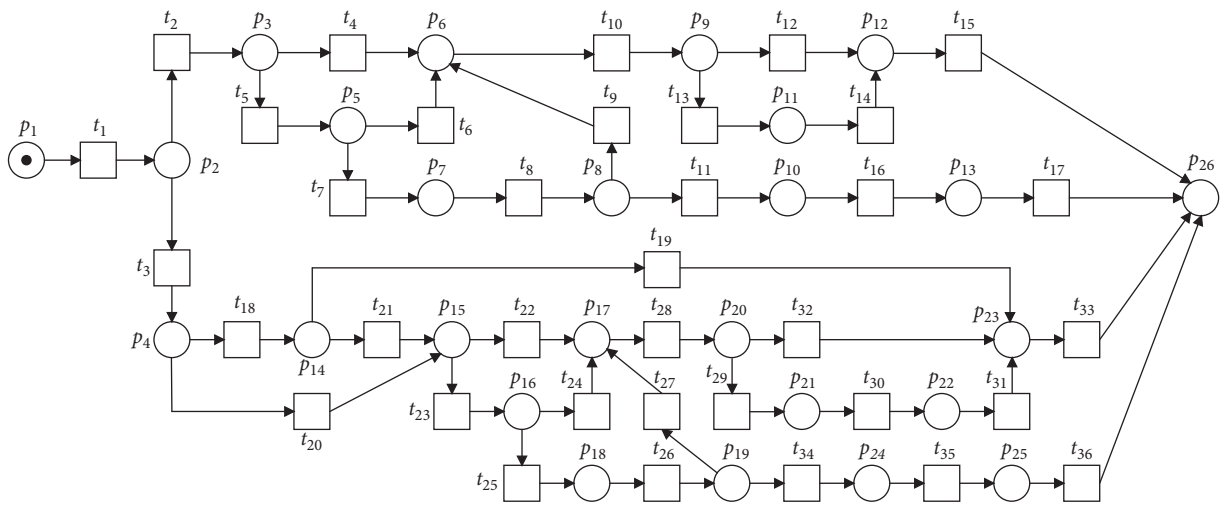

Figure 6: Clinical pathway of STEMI under COVID-19.

comparison of the scale parameters of the two generated models. Figure 9 shows the comparison of the mean queued node number of the two models, and Figure 10 shows the comparison of the mean computation time of the two methods.

This paper does not give a comparison on the quantity of places between the two models because when the trace length and the process model are the same, the quantity of places in the two models is the same, too. Comparing the

Table 2: The relationship between activities and transitions of Figure 6.

Transition	Activity	Transition	Activity
t_1	Medical history/body temperature/ECG/chest CT/blood sample collection	t_{19}	Symptoms mainly with COVID-19
t_2	Exclusion of COVID-19 infection	t_{20}	Stable vital signs
t_3	Suspected or confirmed COVID-19 infection complicated with STEMI	t_{21}	Symptoms mainly with STEMI
t_4	Onset time of STEMI in emergency patients > 12 h	t_{22}	Onset time of STEMI in COVID-19 patients > 12 h
t_5	Onset time of STEMI in emergency patients ≤ 12 h	t_{23}	Onset time of STEMI in COVID-19 patients ≤ 12 h
t_6	Emergency patients with contraindications to thrombolysis	t_{24}	Contraindications to thrombolysis (COVID-19)
t_7	Emergency patients without contraindications to thrombolysis	t_{25}	No contraindications to thrombolysis (COVID-19)
t_8	CCU thrombolysis	t_{26}	Thrombolytic therapy (isolation ward)
t_9	No recanalization of thrombolysis in emergency patients	t_{27}	No recanalization of thrombolysis in COVID-19 patients
t_{10}	Assessment of STEMI patient's benefits and risks	t_{28}	Assessment of STEMI and COVID-19 patient's benefits and risks
t_{11}	Recanalization of thrombolysis in emergency patients	t_{29}	STEMI patient's benefits with COVID-19 > risks
t_{12}	STEMI patient's benefits ≤ risks	t_{30}	Start-up of protection scheme
t_{13}	STEMI patient's benefits > risks	t_{31}	COVID-19 isolation catheter room for PCI
t_{14}	Isolation catheter room for PCI	t_{32}	STEMI patient's benefits with COVID-19 ≤ risks
t_{15}	Continuing treatment in CCU	t_{33}	Isolation ward conservative treatment
t_{16}	CCU conservative treatment	t_{34}	Recanalization of thrombolysis in COVID-19 patients
t_{17}	Selective intervention	t_{35}	Continuing treatment in CCU"
t_{18}	Unstable vital signs	t_{36}	Selective intervention after COVID-19 recovery

Table 3: Experimental environment.

PC model	Lenovo 82JQ
CPU	R7-5800H (3.2 GHz)
Memory	16 GB
OS	Windows10
JDK version	1.8

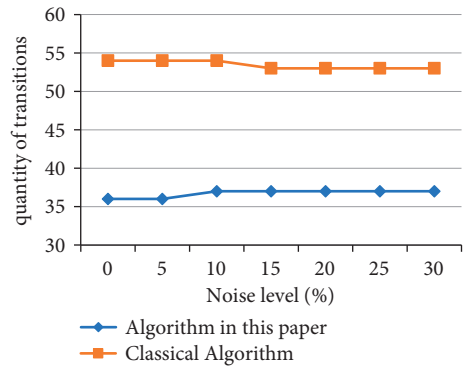

Figure 7: Comparison of the quantity of transitions with different noise levels between two models.

product model in classical algorithm and synchronous composition model in this paper, the quantity of places in the two models is equal to the element number of the union of places in the log model and process model.

Figure 7 compares the quantity of transitions between the two models. The number in the synchronous composition model is about 37, and the number in the product model is about 54. The synchronous composition model reduces some unnecessary transitions, which greatly decreases the scale of the model.

Figure 8 shows the comparison about the quantity of arcs between the two models. The number in the synchronous composition model is about 92, and the number in the product model is about 125. Arc is the flow relationship explained in Definition 2, which indicates the relationship between the transition and the place. The synchronous

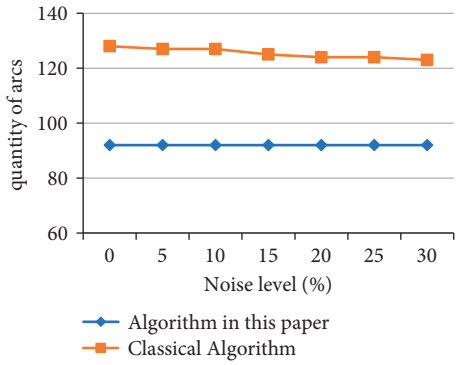

Figure 8: Comparison of the quantity of arcs with different noise levels between two models.

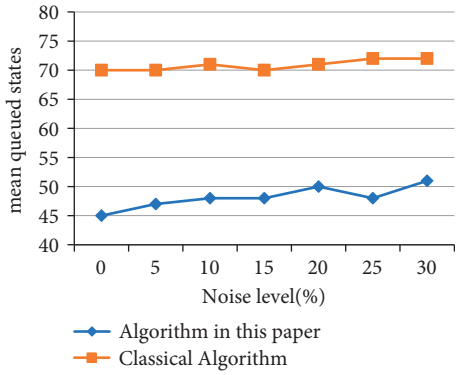

Figure 9: Comparison of the mean queued states (reachable markings) with different noise levels between two models.

composition model not only reduces transitions but also reduces some unnecessary flow relationships, so it reduces the complexity of the search space.

In the part of the optimal alignment computation, Figure 9 shows the quantity of reachable markings in reachable graphs of two models, i.e., mean queued nodes in the optimal alignment computation process. The number in the synchronous composition model is about 47, and the number in the product model is about 72. The quantity of reachable markings affects the time of iteration and sorting efficiency of A $*$ algorithm, as shown in Algorithm 3.

In Figures 7 and 8, the synchronous composition model reduces unnecessary transitions and arcs compared with product model, which makes the reachable graph closer to the real situation and reduces the unnecessary reachable markings.

Figure 10 compares the mean computation time of the two methods in finding the optimal alignment. The mean computation time of Algorithm 3 is about 150 ms, while the time of the classical algorithm is about 245 ms. It saves about 40% of the time and can complete the computation of optimal alignment faster than the classical algorithm at each noise level.

It can be seen from the data that the synchronous composition model used in this paper has lower complexity

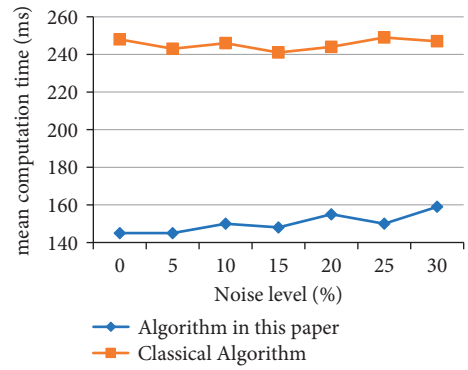

Figure 10: Comparison of the mean computation time with different noise levels between two algorithms.

and higher computational efficiency. Next, the reasons are analyzed and explained.

First of all, the method presented in this paper preprocesses the log when generating the search space and removes noise. This step is also an indispensable step because this method is designed for clinical pathway, which can effectively filter unreasonable noise to prevent the algorithm from failure. Preprocessing is carried out for the log with low time complexity. Even if it cannot be directly reflected from the data, there is a subtle influence on the alignment efficiency and algorithm stability. The product model used in classical method has no need to do this operation because its higher complexity allows it to adapt to more situations at the cost of efficiency. Even if noise can be filtered by such preprocessing, it will not affect the stability of the classical algorithm and has little effect on the efficiency improvement.

Secondly, when generating the search space, the product model will keep the two kinds of original transitions, but the synchronous composition model will not. Assuming that the sum of the transitions in the log model and process model is t, the sum of the arcs is f, and the quantity of synchronous activities is s (assuming that the sum of the flow relationships associated with the log transition and the model transition corresponding to the i-th synchronous activity is f_i); in the product model, the quantity of corresponding transitions is $t + s$, and the quantity of arcs is $f + \sum_{i=1}^{s} f_i$; in the synchronous composition model, the quantity of transitions is $t - x$, and the quantity of arcs is f. That is, each synchronous activity will make product model have 2 more transitions and several arcs than the synchronous composition model. These additional transitions and arcs have no actual meaning. The preprocessing step of this method solves the problem in advance and simplifies the scale of the model as much as possible. Hence, the search space of this method is much smaller than that of the classical method. In addition, the higher the noise level, the lower the number of synchronous activities. Figures 7 and 8 show that with the gradual increase of the noise level, the scale of the two models is gradually close. However, the higher the noise ratio is, the closer the trace is to the real noise, the lower the significance of the analysis is, and the more it should be abandoned.

Table 4: The relationship between transitions and activities of trace σ.

Transition	Label	Activity
t_2	e	Exclusion of COVID-19 infection
t_8	c	CCU thrombolysis
	t	Transfer to isolation ward
t_{11}	r	Recanalization of thrombolysis in emergency patients

In the part of computing the optimal alignment, both methods use A $*$ search algorithm. The time complexity of A $*$ search algorithm in Algorithm 3 is $O(n^2\log_2{}^n)$, and n is affected by the quantity of reachable markings. Even if the difference of the scale between the two models is not large, there will still be a large efficiency gap in the two methods. Figures 7–10 show that the quantity of transitions and arcs decreases less than 30%, and the mean queued nodes reduces by about 34%, and the mean computation time of optimal alignment reduces by about 40%.

According to the data and analysis in the experiments, when calculating the optimal alignment, since the efficiency of the optimal alignment increases geometrically with the model complexity, the selection of synchronous composition model can greatly improve the alignment efficiency.

4.4. Algorithm Effectiveness Verification. After the computation of the optimal alignment is completed, the optimal alignment results such as formula (2) are obtained. It can be clearly seen from the equation whether the activities of the patient's diagnosis and treatment process are normal and whether there are missing or multiple activities. Therefore, the medical staff can master the whole clinical pathway diagnosis and treatment process. An example in the diagnosis and treatment log is illustrated below.

Trace $\sigma_2 = <\ldots$, exclusion of COVID-19 infection, $\ldots$, CCU thrombolysis, transfer to isolation ward, recanalization of thrombolysis in emergency patients, $\ldots>$, , and here, part of synchronous transitions are replaced by ellipses. The relationship between the activities and the transitions is shown in Table 4.

Trace σ_2 is denoted as $<\ldots, e, \ldots, c, t, r, \ldots>$ according to Table 4, and the optimal alignment result is shown in the following:

$$\gamma_2 = \begin{array}{|c|c|c|c|c|c|c|} \hline \ldots & e & \ldots & c & t & r & \ldots \\ \hline \ldots & e & \ldots & c & >> & r & \ldots \\ & t_2 & & t_8 & & t_{11} & \\ \hline \end{array} \quad (3) \qquad (3)$$

According to the optimal alignment result of formula (3), it can be seen that the patient was excluded from COVID-19 infection at the beginning, but later transferred to the isolation ward which may be caused by many reasons. For example, as the epidemic situation intensifies, it is necessary to strengthen the protective measures for high-risk patients; or the patient contacted outside visitors with risk of infection during treatment, or the patient has concealed the travel information before admission to the hospital. The specific situation needs to be analyzed by the medical staff with the actual situation, which cannot be judged only from the logs.

The classical algorithm has been widely recognized and used for a long time, and its effectiveness is beyond reasonable doubt. Hence, the optimal alignments computed by the two algorithms need to be compared one by one. If all the results are the same, Algorithm 3 can be considered to be effective. Otherwise, if the results are different, it is possible that this algorithm be considered to be invalid.

Since the optimal alignment result is a sequence composed of nodes in the form of $(\alpha_2(t_x'), (\alpha_1(t_y), t_y))$, where t_x' corresponds to log model transition, $\alpha_2(t_x')$ is the activity corresponding to t_x', t_y corresponds to process model transition, and $\alpha_1(t_y)$ is the activity corresponding to t_y. Hence, after the optimal alignment is completed, the optimal alignment results obtained by the two algorithms are compared one by one to check whether the activity of the log model and the activity and the transition of the process model are exactly the same. According to the comparison results, it shows that all the optimal alignments are the same, which can illustrate that the algorithm is effective in deviation detection.

5. Conclusions

The deviation detection of clinical pathway is an important technology to standardize and study the diagnosis and treatment process and optimize the clinical pathway. The classical algorithm presented by Adriansyah et al. can synthesize the trace and process model and has a wide range of applications. Compared with the previous forward-looking methods, it has made a qualitative leap. However, with high complexity of search space, the efficiency in computing the optimal alignment is still unsatisfactory. Hence, this paper presents an optimization algorithm.

In this paper, the diagnosis and treatment log is preprocessed by the particularity of the diagnosis and treatment process. Considering that the alignment is logical based on the reachable graph to find the shortest path, even if the unnecessary transitions are only slightly reduced, the alignment efficiency can still be greatly improved. Hence, this paper uses the synchronous composition model for alignment computation, which greatly reduces the alignment time.

Finally, the clinical pathway of STEMI under COVID-19 is taken as an example for experimental analysis. It is illustrated that the newly proposed deviation detection method has the advantages of higher efficiency compared with the traditional algorithm, which can greatly shorten the time consumed in the deviation detection process. In the future work, this method can be applied to online deviation detection. Online deviation detection can timely detect the anomaly and remind patients of the deviation existing in the current completed diagnosis and treatment process. Meanwhile, it can predict the possible subsequent deviation and provide judgment basis for medical staff to carry out manual intervention to avoid adverse events. Online

deviation detection requires higher efficiency, which can better reflect the advantage of the new method.

Data Availability

The data used to support the findings of this study are openly available at https://drive.google.com/file/d/1gkPRQXDNBzcAkoIcLdFLGMOGKluPhVdf/view?usp=sharing.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This study was supported in part by the National Natural Science Foundation of China under grant nos. 61973180 and 72101137, in part by the Tai'shan Scholar Construction Project of Shandong Province, in part by the Education Ministry Humanities and Social Science Research Youth Fund Project of China under grant nos. 20YJCZH159 and 21YJCZH150, in part by the Natural Science Foundation of Shandong Province under grant nos. ZR2019MF033, ZR2020MF033, and ZR2021MF117, and in part by the Shandong Key R&D Program (Soft Science) Project under grant no. 2020RKB01177.

References

[1] S. Chu, "Reconceptualising clinical pathway system design," *Collegian*, vol. 8, no. 1, pp. 33–36, 2001.

[2] T. J. Marrie, C. Y. Lau, S. L. Wheeler, C. J. Wong, M. K. V. andervoort, and B. G. Feagan, "A controlled trial of a critical pathway for treatment of community-acquired pneumonia," *Journal of the American Medical Association*, vol. 283, no. 6, pp. 749–755, 2000.

[3] X. Y. He, M. K. Bundorf, J. J. Gu, P. Zhou, and D. Xue, "Compliance with clinical pathways for inpatient care in Chinese public hospitals," *BMC Health Services Research*, vol. 15, no. 1, p. 459, 2015.

[4] L. Wen, J. Wang, W. M. P. van der Aalst, B. Huang, and J. Sun, "Mining process models with prime invisible tasks," *Data and Knowledge Engineering*, vol. 69, no. 10, pp. 999–1021, 2010.

[5] C. Liu, H. Duan, Q. Zeng, M. Zhou, F. Lu, and J. Cheng, "Towards comprehensive support for privacy preservation cross-organization business process mining," *IEEE Transactions on Services Computing*, vol. 12, no. 4, pp. 639–653, 2019.

[6] C. Wong, F. Visram, D. Cook et al., "Development, dissemination, implementation and evaluation of a clinical pathway for oxygen therapy," *Canadian Medical Association Journal*, vol. 162, no. 1, pp. 29–33, 2000.

[7] T. A. Brennan, L. E. Hebert, N. M. Laird et al., "Hospital characteristics associated with adverse events and substandard care," *Journal of the American Medical Association*, vol. 265, no. 24, pp. 3265–3269, 1991.

[8] Y. Tian, X. P. Ma, X. G. Zhang, and T. Zhang, "Evaluation and optimization of clinical pathway based on Petri net," *Computer Science*, vol. 40, no. 5, pp. 193–197, 2013.

[9] A. Rodríguez, A. Caro, C. Cappiello, and I. Caballero, "A BPMN extension for including data quality requirements in business process modeling," *Lecture Notes in Business Information Processing*, vol. 125, pp. 116–125, 2012.

[10] V. Huser, "Book review," *Journal of Biomedical Informatics*, vol. 45, no. 5, pp. 1018-1019, 2012.

[11] W. M. P. van der Aalst, "Formalization and verification of event-driven process chains," *Information and Software Technology*, vol. 41, no. 10, pp. 639–650, 1999.

[12] C. Liu, Q. Zeng, H. Duan, M. Zhou, F. Lu, and J. Cheng, "E-net modeling and analysis of emergency response processes constrained by resources and uncertain durations," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 45, no. 1, pp. 84–96, 2015.

[13] L. Qi, M. Zhou, and W. Luan, "Impact of driving behavior on traffic delay at a congested signalized intersection," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 7, pp. 1882–1893, 2017.

[14] L. Qi, M. Zhou, and W. Luan, "A two-level traffic light control strategy for preventing incident-based urban traffic congestion," *IEEE Transactions on Intelligent Transportation Systems*, vol. 19, no. 1, pp. 13–24, 2018.

[15] S. Kansal and P. Dabas, "An introduction to signed Petri net," *Journal of Mathematics*, vol. 2021, Article ID 5595536, 8 pages, 2021.

[16] Y. Du, L. Qi, and M. Zhou, "Analysis and application of logical Petri nets to E-commerce systems," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 44, no. 4, pp. 468–481, 2014.

[17] J. Cheng, C. Liu, M. Zhou, Q. Zeng, and A. Yla-Jaaski, "Automatic composition of semantic web services based on fuzzy predicate Petri nets," *IEEE Transactions on Automation Science and Engineering*, vol. 12, no. 2, pp. 680–689, 2015.

[18] Y. Du, L. Wang, and M. Qi, "Constructing service clusters based on service space," *International Journal of Parallel Programming*, vol. 45, no. 4, pp. 982–1000, 2017.

[19] X. Cong, M. P. Fanti, A. M. Mangini, and Z. Li, "Decentralized diagnosis by Petri nets and integer linear programming," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 48, no. 10, pp. 1689–1700, 2018.

[20] D. M. Xiang, G. J. Liu, C. G. Yan, and C. G. Jiang, "Detecting data-flow errors based on Petri nets with data operations," *IEEE/CAA Journal of Automatica Sinica*, vol. 5, no. 1, pp. 251–260, 2018.

[21] Y. Teng, Y. Du, L. Qi, and W. Luan, "A logic Petri net-based method for repairing process models with concurrent blocks," *IEEE Access*, vol. 7, pp. 8266–8282, 2019.

[22] Y. Y. Du, C. J. Jiang, M. C. Zhou, and Y. Fu, "Modeling and monitoring of E-commerce workflows," *Information Sciences*, vol. 179, no. 7, pp. 995–1006, 2008.

[23] Y. Y. Du, C. J. Jiang, and M. C. Zhou, "A Petri net-based model for verification of obligations and accountability in cooperative systems," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems and Humans*, vol. 39, no. 2, pp. 299–308, 2009.

[24] Y. Du, L. Qi, and M. Zhou, "A vector matching method for analysing logic Petri nets," *Enterprise Information Systems*, vol. 5, no. 4, pp. 449–468, 2011.

[25] S. Ye, J. H. Ye, and C. C. Liu, "Algorithm for finding the critical paths based on Petri net," *Computer Science*, vol. 39, no. 6, pp. 201–203+221, 2012.

[26] J. Li, R. Yang, Z. J. Ding, and M. Pan, "A method for learning a Petri net model based on region theory," *Computing and Informatics*, vol. 39, no. 1-2, pp. 174–192, 2020.

[27] S. Shahzadi, X. W. Fang, and D. A. Alilah, "Role of Stochastic Petri Net (SPN) in process discovery for modelling and analysis," *Mathematical Problems in Engineering*, vol. 2021, Article ID 8699164, 7 pages, 2021.

[28] Y. H. Tian, Y. Y. Du, D. Han, and W. Liu, "Alignments between batch traces and process models based on Petri nets," *Chinese Journal of Computers*, vol. 41, no. 3, pp. 611–627, 2018.
[29] X. W. Feng, D. Han, and Y. H. Tian, "Analysis and application of min-cost transition systems to business process management," *Computing and Informatics*, vol. 39, no. 1-2, pp. 213–245, 2020.
[30] A. Rozinat and W. M. P. V. D. Aalst, "Conformance checking of processes based on monitoring real behavior," *Information Systems*, vol. 33, no. 1, pp. 64–95, 2007.
[31] D. Han and Y. H. Tian, "Analysis and application of transition systems based on Petri nets and relation matrices to business process management," *Mathematical Problems in Engineering*, vol. 2020, Article ID 2545413, 18 pages, 2020.
[32] A. Adriansyah, *Aligning Observed and Modeled Behavior*, Eindhoven University of Technology, Eindhoven, Netherlands, 2014.
[33] J. E. Cook and A. L. Wolf, "Software process validation," *ACM Transactions on Software Engineering and Methodology*, vol. 8, no. 2, pp. 147–176, 1999.
[34] W. Song, X. Xia, H. A. Jacobsen, P. Zhang, and H. Hu, "Efficient alignment between event logs and process models," *IEEE Transactions on Services Computing*, vol. 10, no. 1, pp. 136–149, 2017.