Hindawi
Journal of Advanced Transportation
Volume 2021, Article ID 2869521, 17 pages
https://doi.org/10.1155/2021/2869521

Research Article

Causal Discovery of Flight Service Process Based on Event Sequence

Qian Luo,[1,2] Lin Zhang,[1,2] Zhiwei Xing,[1,2] Huan Xia,[1] and Zhao-Xin Chen[1]

[1]*The Second Research Institute of Civil Aviation Administration of China, Chengdu 610041, China*
[2]*School of Electronic Information and Automation, Civil Aviation University of China, Tianjin 300300, China*

Correspondence should be addressed to Zhiwei Xing; cauc_xzw@163.com

Received 27 April 2021; Revised 22 July 2021; Accepted 19 August 2021; Published 18 September 2021

Academic Editor: Rocío de Oña

The development of the civil aviation industry has continuously increased the requirements for the efficiency of airport ground support services. In the existing ground support research, there has not yet been a process model that directly obtains support from the ground support log to study the causal relationship between service nodes and flight delays. Most ground support studies mainly use machine learning methods to predict flight delays, and the flight support model they are based on is an ideal model. The study did not conduct an in-depth study of the causal mechanism behind the ground support link and did not reveal the true cause of flight delays. Therefore, there is a certain deviation in the prediction of flight delays by machine learning, and there is a certain deviation between the ideal model based on the research and the actual service process. Therefore, it is of practical significance to obtain the process model from the guarantee log and analyze its causality. However, the existing process causal factor discovery methods only do certain research when the assumption of causal sufficiency is established and does not consider the existence of latent variables. Therefore, this article proposes a framework to realize the discovery of process causal factors without assuming causal sufficiency. The optimized fuzzy mining process model is used as the service benchmark model, and the local causal discovery algorithm is used to discover the causal factors. Under this framework, this paper proposes a new Markov blanket discovery algorithm that does not assume causal sufficiency to discover causal factors and uses benchmark data sets for testing. Finally, the actual flight service data are used for causal discovery among flight service nodes. The local causal discovery algorithm proposed in this paper has a certain competitive advantage in accuracy, F1, and other aspects of the existing causal discovery algorithm. It avoids the occurrence of its dimensional disaster. Through the in-depth analysis of the flight safety reason node discovered by this method, it is found that the unreasonable scheduling of flight support personnel is an important reason for frequent flight delays at the airport.

1. Introduction

The 2019 Civil Aviation Industry Development Statistics Bulletin [1] shows that compared with 2018, the total civil aviation transportation turnover in 2019 has increased by 7.2%. The increase in the total airport transportation turnover requires the improvement of airport flight service efficiency. Flight service includes various activities, such as opening the cabin door, cleaning, and refueling. The flight service process is highly dependent, and the delay of a single node will affect the delay of subsequent operations, resulting in delays in the launch of flights. These attributes add additional complexity to the assurance operation. For example, to add aviation fuel for a departing flight, we need to know the type of aircraft of this flight, the position of the aircraft, the planned departure time of the aircraft, and other decision-making information, in order to determine when, where, and how much aviation fuel the tanker will carry to complete the flight. It is a complicated process in itself. However, its own attributes, such as the fuel capacity and integrity of the scheduled aviation fuel vehicle, will also affect whether the aviation fuel addition can be completed normally. The attributes of the service equipment itself make the work of aviation fuel addition more complicated. In view of the highly dependent characteristics and complexity of the flight service process mentioned above, process managers usually do not know which operation nodes have problems during flight service operations that will ultimately affect the overall performance of the flight service, such as flight departure delays, the duration of refueling truck operations, etc. Therefore, when an airport is experiencing delays in the launch of departing flights, it is difficult for airport managers to find the service nodes that directly cause the delay and take targeted improvement measures to address the problem nodes.

Some scholars [2–5] have established flight service process models with methods such as critical path restoration, colored Petri nets, and service model tools for the flight service process. Based on these models, they conducted in-depth studies on the scheduling of ground service resources. But the above studies are driven by a manually drawn model. This is an ideal model, which is deviated from the actual support business process model. The deviation is mainly reflected in the manually drawn ideal model. The node importance and topological sorting of each support node are only based on expert knowledge. As the managers of each airport are different, and their specific service plans have their own characteristics, the artificially drawn aircraft service model is not completely applicable to the flight service process of a specific airport. At the same time, the research on flight launch delays focuses on predicting the length of the delay, while the prediction method based on statistical correlation emphasizes the correlation between influencing factors and results rather than causality, so it is impossible to analyze which links ultimately lead to the flight launch delay, And, due to the existence of unobservable variables, there is a certain deviation in the prediction results.

At present, flight support operations are supported by airport information systems. These systems record the history of outbound flight support operations in the form of event sequences. Therefore, in order to analyze the performance of flight support, the problem of deviations in artificially established process models are to be avoided. This paper uses process mining to dig out the actual process model of outbound flight services from the sequence of events. Process mining is a tool that transforms event data into business insights. It bridges the gap between traditional model-based process analysis (such as simulation and other business process management techniques) and data-centric analysis techniques (such as machine learning and data mining) [6]. Through process mining, we get a practical business process model, and on this basis, the performance indicators of each link are calculated to describe the operation status of each link. By analyzing the causal relationship between the performance indicators of each link and the flight departure delay, we can determine which node's operation situation affects the flight departure and launch, and provide effective guidance for reducing flight delays and improving airport operation efficiency.

In this existing flight support research, no research about causal factors of flight delay has been done in business processes based on actual service logs. This paper proposes a framework to realize automatic discovery of causal factors in the presence of latent causal variables based on process mining (ACLP), the flight service process model mined by the process mining algorithm, and the premise of relaxing the sufficiency of causality, using the score-based maximum ancestor graph Markov blanket. The algorithm (SMMB) generates a local ancestor map of the flight service launch delay and uses the direction between nodes extracted by the process model as a supplement to the local ancestry graph to adjust the direction of the edge and to realize the automatic discovery of cause and effect of the flight service business performance. Our proposed framework combines the process mining method with the causality discovery method under the existence of unobservable variables. When the business process of flight service is unknown, a flight service process model that fits the actual situation is established based on the sequence of events with process mining. Aiming at the problem of unobservable variables in actual scenarios, the newly proposed SMMB algorithm is used to determine the causal relationship between business process performance indicators, and the extraction of causal factors from event data to explaining business process performance is realized. The SMMB algorithm proposed in this paper is based on the score-based local directed acyclic graph (DAG) discovery algorithm, that is, the score-based local learning (SLL) algorithm, which is extended according to the characteristics of the maximal ancestral graph (MAG) Markov blanket (MB). It is a topology-based method. The neighbor set and spouse set of the target variable is constructed by the method of scoring. Then, according to the relevant definition and inference of the area set proposed by [7], the adjacent area set is determined, and the complete MB is searched. Through the advantages of the scoring method in searching the neighborhood set and spouses set of the target variable, the SMMB algorithm has a better performance on the *F*-measure evaluation index than the constraint-based MAG MB algorithm proposed before. It provides new ideas for the automatic causal discovery of the flight service process under latent variables. This article is organized as follows: Section 2 discusses the background, Sections 3 and 4, respectively, introduce the proposed framework and experimental results and give some suggestions based on the experimental results. Section 5 draws conclusions and makes future work outlook.

In this paper, a new framework is used to discover the root cause of business process performance problems automatically. Compared with the automated causal discovery of business process performance based on the Granger causality test proposed by some researchers, the contribution of the method proposed in this paper is as follows: (1) It is the first time that an automatic causal discovery method for flight support that combines the MAG algorithm and the process mining algorithm has been proposed. The MAG algorithm is used to replace the Granger causality test used in the previous process causality discovery method for causal discovery to avoid potential confounding effects. Bidirectional edges are marked as the node pairs with latent variables. (2) The SMMB local causality discovery algorithm is proposed to realize the search for flight delays with higher accuracy and avoid the curse of dimensionality.

2. Related Work

2.1. Process Model Discoveries. Whether defined and prescribed or implicit and temporary, business processes drive and support most of the functions and services in today's world's enterprises and management organizations. Based on the complexity of process control flow and the related concepts of process repeatability and predictability, the research of Ciccio et al. [8] divides business processes into the following three macro types: structured processes, semistructured processes, and unstructured processes. The structured process is characterized by a clearly defined, predictable, and repeatable sequence of activities and its input and output are predefined, while the semistructured and unstructured process has no predefined and repeatable sequence of activities. The semistructured process can outline the possible sequence of activities based on the case, determine the input of the required activities, and change the sequence of some activities through a specific situation's characteristics. The activities of the unstructured process are differently combined based on the specific instance. The sequence of activities becomes completely case-dependent when the level of process flexibility and unpredictability is increased.

Process mining is a method of analyzing actual business processes based on event logs generated by the system. The idea is to discover, monitor, and improve real business processes by extracting knowledge from event logs.

Definition 1 (event [9]). An event is the instantiation of an activity in a business process, usually represented by a tuple $e = (a, \text{caseID}, t_{\text{start}}, t_{\text{end}}, k_1, \ldots, k_m)$, where a represents the activity name attribute corresponding to the event, caseID represents the instance attribute where the event is located, eventID represents the event ID attribute of the event, t_{start} represents the start timestamp attribute of the event, t_{end} represents the end timestamp attribute of the event, and $k_1, \ldots, k_m\,(m \geq 0)$ represents other attribute values, where $\forall i \in [1, m]$, $k_i \in K_i$, K_i represent the value range of each attribute. The event log L for a specific process model comprises a series of events in the process instance. The sequence of all events in the process instance in chronological order is the trajectory. A complete trajectory corresponds to one execution of the process. In terms of flow, all historical execution traces constitute the event log L.

The discovery of process models has always been a hot issue in process mining. In the absence of prior knowledge, information about the original process model, organizational context, and execution properties can be obtained from the execution log. Most process discovery algorithms usually apply a single algorithm to control flow steps [10], such as alpha algorithm [11], heuristic mining algorithm [12], multistage process mining algorithm [13], and region-based mining algorithm [14]. The above algorithm is effective when applied to a structured business process. Still, when applied to a semistructured or unstructured process, the model found by the above algorithm is really "spaghetti-like." These models describe every detail of unstructured behavior found in logs too finely. The reason for the problem lies in the assumptions that these process mining algorithms are based on. Assume as follows:

Assumption 1 [14]: all logs are reliable and trustworthy

Assumption 2 [14]: there exists an exact process which is reflected in the logs

These assumptions are completely reasonable in a structured and controlled environment, but they do not hold true in a less structured real environment. Therefore, the process model discovery algorithm based on the above assumptions will simulate the entire process completely, accurately, and meticulously. The results are often "spaghetti," and process managers cannot obtain effective information from the model.

In order to solve the above problems, Christian et al. [15] proposed a process mining algorithm based on fuzzy theory. When dealing with unstructured problems, the fuzzy algorithm can distinguish whether the task is important or not and can remove unnecessary details. A more advanced view is abstracted, which focuses on discovering a more advanced mapping of behavior in the log rather than trying to discover the true process model.

2.2. Causal Discovery Algorithm. Let $V = \{V_1, \ldots, V_N\}$ contain N observed variables, P be a discrete joint probability distribution over V, and G represent DAG. We call the triple $\langle P, V, G\rangle$ a Bayesian network (BN), if $\langle P, V, G\rangle$ satisfies the Markov condition: each variable is independent of any subset of its nondescendant variables conditioned on its parents in G. The causal relationship between variables in BN can be represented by DAG G containing only directed edges ($\longrightarrow$).

Definition 2 (causal sufficiency [7]). The observed variable set V is said to be causal sufficient if and only if any common cause of two or more variables in V is also in V.

Causal sufficiency considers that given a set of observed variables V, there is no latent common cause for V's subset of variables.

Definition 3 (faithfulness [16]). In a BN$\langle P, V, G\rangle$, G is faithful to the probability distribution P over V if and only if every independence present in P is entailed by G and Markov conditions. P is faithful if and only if G is faithful to P.

The faithfulness assumption establishes a relationship between the probability distribution P and its underlying DAG G. We can use a conditional independence test instead of d separation to find all BN's dependencies or independence under this assumption. Under the assumption of satisfying the sufficiency of causality, the MB of the target variable in the DAG includes the parents, children, and spouses of the target variable T. Nowadays, the MB discovery algorithm for DAG has been relatively complete. It can be divided into topology-based methods and nontopological methods. The nontopological methods greedily test each variable and target by using the definition of Markov blanket, like the IAMB algorithm [17]. The topology-based approach aims to gradually search for the MB of the target node, such as Min-Max Markov Blanket (MMMB) [18], using the topological characteristics of the MB. Articles [19, 20] introduce the same framework based on the topology method and conduct extensive experimental research to verify its superior performance in various applications.

Without assuming the sufficiency of causality, when the underlying data generated have potential common causes, MAG is proposed to represent the independent relationship between the observed variables. There is no need to mark the potential common causes in the structure clearly. A hybrid graph is a collection of nodes and edges, and its edges may be one-way edges ($\longrightarrow$) or bidirectional edges ($\longleftrightarrow$). Suppose there is no directed ring (the presence of $V_i \longrightarrow V_j$ and $V_i \longleftarrow V_j$ at the same time) and almost directed ring (the presence of both $V_i \longleftrightarrow V_j$ and $V_i \longleftarrow V_j$) in the mixed graph, it is called an ancestor graph. On the path τ of the ancestry graph, if path τ contains $* \longrightarrow V_i \longleftarrow *$, then the nonendpoint variable V_i is the colliding node in the ancestry graph. Otherwise, V_i is a noncolliding node on τ. Every nonendpoint variable on the collision path from the target node T to Y in the MAG is a collision node. For example, $T \longleftrightarrow V_1 \longleftrightarrow V_2 \longleftrightarrow V_3 \longleftrightarrow Y$ is a collision path, and all V_1, V_2, V_3 are collision nodes.

Definition 4 (V-structure [7]). In an ancestry graph, a triple $\{V_i, V_j, V_k\}$ is an unshielded triple if V_i and V_j are adjacent, and V_j and V_k are adjacent, but V_j and V_k are not adjacent. An unshielded triple $\{V_i, V_j, V_k\}$ is called a v-structure if V_j is a collider on the path $\{V_i, V_j, V_k\}$, and the triple satisfies $\exists Z \subseteq V \backslash \{V_i, V_j, V_k\}$ such that $V_i \perp\!\!\!\perp V_j | Z$ and $V_i \perp\perp V_j | \{Z \cup V_j\}$. In a v-structure, $V_i \longrightarrow V_j \longleftarrow V_k$, V_k is the spouse node of V_i.

Definition 5 (m-connection and m-separation [21]). In the ancestor graph $\mathrm{G} = (E, V)$, given a set of nodes $Z, Z \subseteq V \backslash \{A, B\}$, if it satisfies the following: (1) the noncolliding nodes on the path p do not belong to Z and (2) each colliding node on the path is the ancestor of a member of Z, so the path p between A and B is m-connection. If there is no m-connection path concerning Z between A and B, then A and B are m-separation.

Definition 6 (maximum ancestor graph [22]). For any two nonadjacent variables in an ancestor graph, if there is a set of variables m-separating them, the ancestor graph is maximal.

There are relatively few discovery algorithms for MAG MB. The article [7] proposed for the first time the constraint-based local causality discovery algorithm (M3B algorithm) of MB under the MAG framework that does not assume the sufficiency of causality, instead of learning the overall MAG and directly learning MB. This algorithm is a topology-based MB algorithm. The algorithm first finds the neighborhood set (parents and children) of the target node and uses a recursive search algorithm to recursively find the area set of a given target to complete the MB. The article [22] has proved that the constraint-based method is sensitive to error propagation, and there is no scoring method for the algorithm of MAG MB discovery. Therefore, the framework of the SLL algorithm mentioned is extended according to the characteristics of MAG MB. First, the neighbor set and spouse set of the target variable are constructed by the method of scoring. Then, according to the relevant definition and inference of the area set proposed in the literature [23], the adjacent area set of the target node is determined, and the MB is finally completed.

This paper optimizes on the basis of the fuzzy process model and obtains the actual flight guarantee process model. Based on the actual business process, the performance indicators of the nodes are calculated to measure the operating status of each node, and finally, the SMMB local causality mining algorithm is used to find the root cause of the delay in flight launch when latent variables exist. This method provides new research ideas for discovering causal factors in process mining in the presence of latent variables. For the automatic discovery of causal factors of business performance, few scholars have done in-depth research. For example, literature [24, 25] proposed a method for automatically discovering process performance bottlenecks and deviations based on event data, but it did not explore causality. Literature [26] proposed a method based on time series analysis to detect the causal relationship between business process characteristics and process performance indicators. However, the Granger causality test method adopted did not consider the existence of latent variables; that is, it believed that the actual data satisfy the assumption of causality sufficiency. Therefore, this article's contributions are as follows: (1) For the first time, it is proposed to realize the automatic discovery of causal factors of business performance under the premise of relaxing the assumption of causal adequacy. (2) The proposed local MAG discovery algorithm based on scoring has more advantages than M3B and RFCI algorithms.

3. Proposed Framework

This section proposes a framework to realize the automatic discovery of flight guarantee causality from the sequence of the events, as shown in Figure 1. The framework consists of two parts: (1) process model mining and (2) construction of the local causal structure. The first part is the fuzzy mining algorithm, which mainly includes two stages. The first stage is the initialization stage, which establishes the initial process model through the flight guarantee event sequence. The second stage is the simplified stage, which mainly includes three parts, conflict resolution, edge filtering, clustering, and abstraction. This article proposes solutions to the unary and N-ary conflicts in the initial model and optimizes the process model. The second part is the local causal construction. This method is based on the SLL algorithm. It is mainly used to search for the neighboring nodes and spouse nodes of the target node and combines MAG MB based on the searched neighboring node set and spouse set. The feature searches the area set of the target node, the parent set of the area set, etc., and then completes the complete Markov blanket's construction.

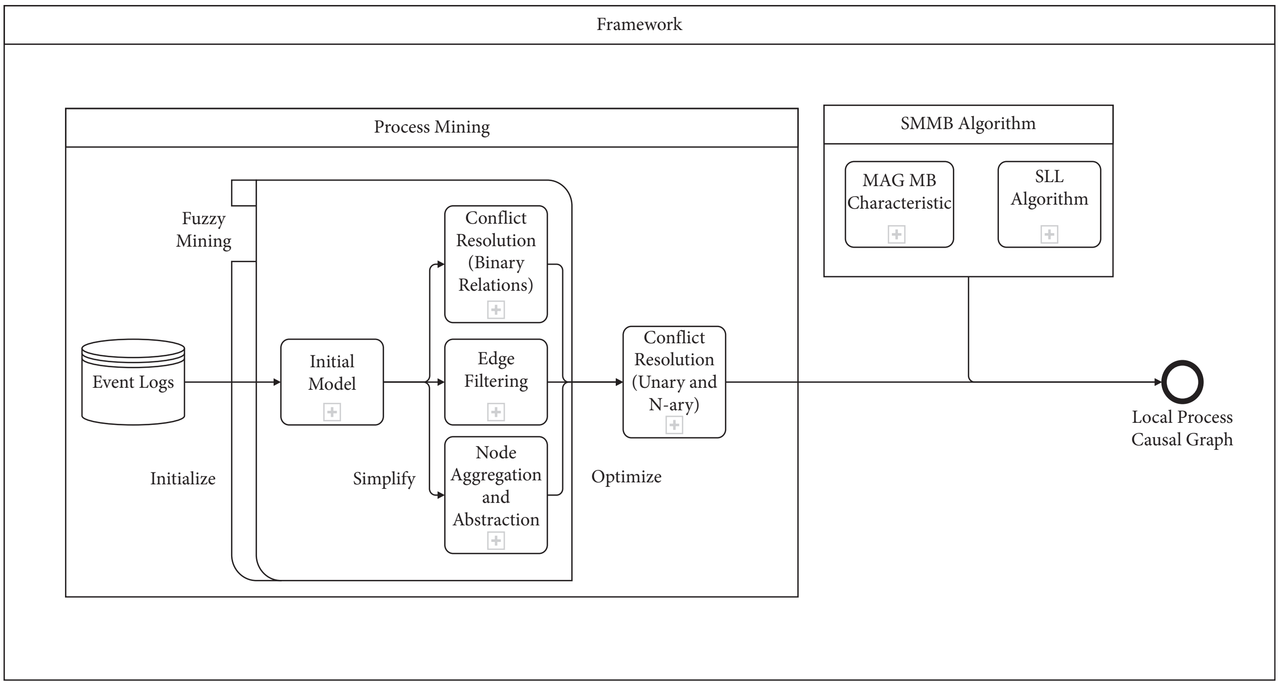

Figure 1: Automatic discovery framework of causal factors based on process mining.

3.1. Process Model Mining. As mentioned earlier, the fuzzy mining algorithm is divided into two stages. The first stage is the initialization stage, which converts each observed event type into an activity node. The directed edges added to the model represent the ordering relationship between activities. The second stage is the simplification stage, divided into three steps: conflict resolution of binary relations, edge filtering, aggregation, and abstraction. As shown in Figure 2, the initialization model's possible conflict relationships include binary conflicts, N-ary conflicts, and unary conflicts. The conflict resolution part of the fuzzy mining algorithm only includes the solution of the binary conflict problem, which leads to the phenomenon of the N-ary cycle and self-circulation in the process model obtained by the fuzzy process mining algorithm. Self-circulation is possible in the flight service process. Take the change of aircraft parking position as an example. When the aircraft enters the airport, due to the shortage of aircraft space resources, the aircraft needs to change its parking position several times. Therefore, in the event sequence, the change of aircraft parking position will appear multiple times in a row, and the model obtained by the process mining algorithm will have a self-circulation phenomenon. The existence of this phenomenon may be a pure exception. There are no binary and N-ary cycles in the actual operation process of flight service, and it operates in sequence over time. Aiming at the reason for this phenomenon and the solution of binary conflict of fuzzy mining algorithm, the solution of N-ary conflict and unary conflict is derived to optimize the process model of fuzzy mining.

In the fuzzy mining algorithm [27], the generation of binary conflicts is divided into three situations: binary loop, exception, and concurrency. If the relative importance $\mathrm{rel}(A,B)$ and $\mathrm{rel}(B,A)$ of two conflicting relationships exceed the retention threshold, then activities A and B form a binary cycle. If at least one conflicting relationship is lower than this threshold, determine the offset between the relative importance, $S(A,B)=\mathrm{rel}(A,B)-\mathrm{rel}(B,A)$. If the offset value exceeds the ratio threshold, the less important relationships are deleted. Suppose at least one relationship wants to retain the threshold for importance, and the offset value is less than the ratio threshold. In that case, the relationship between A and B is a low and balanced relationship, implying that A and B are executed at the same time, so both edges are deleted at the same time. The formula for relative importance is as follows:

$$\mathrm{rel}(A,B)=\frac{1}{2}\times\frac{\mathrm{sig}(A,B)}{\sum_{X\in\eta}\mathrm{sig}(A,X)}+\frac{1}{2}\times\frac{\mathrm{sig}(A,B)}{\sum_{X\in\eta}\mathrm{sig}(X,B)},\qquad(1)$$

where η is the node set of the process model, sig: $\eta\times\eta\longrightarrow R_0^+$ is the priority relationship assigned to each pair of nodes $A,B\in\eta$, and rel: $\eta\times\eta\longrightarrow R_0^+$ is the relative importance between each pair of nodes A and B.

On the basis of the solution of the binary conflict, the solution of the N-ary conflict is derived. First, the relative importance of the binary relationship does not apply to the N-ary conflict, so it needs to be expanded to the relative importance of the N-ary relationship.

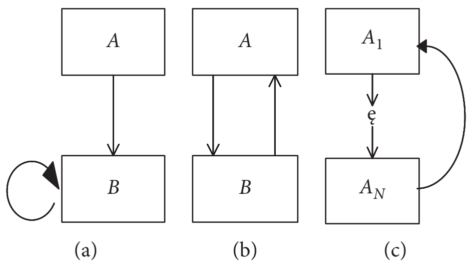

Figure 2: Conflicting relationships in the initial model. (a) Unary conflict. (b) Binary conflict. (c) N-ary conflict.

$$\begin{aligned}\mathrm{rel}_i\left(A_i, A_{i+1}, \ldots, A_N, A_1, \ldots, A_{i-1}\right) = {} & \frac{1}{2(N-1)} \times \frac{\mathrm{sig}\left(A_i, A_{i+1}\right)}{\sum_{X \in \eta} \mathrm{sig}\left(A_i, X\right)} + \frac{1}{2(N-1)} \times \frac{\mathrm{sig}\left(A_i, A_{i+1}\right)}{\sum_{X \in \eta} \mathrm{sig}\left(X, A_{i+1}\right)} \\ & + \frac{1}{2(N-1)} \times \frac{\mathrm{sig}\left(A_i, A_{i+1}\right)}{\sum_{X \in \eta} \mathrm{sig}\left(A_{i+1}, X\right)} + \cdots + \frac{1}{2(N-1)} \times \frac{\mathrm{sig}\left(A_N, A_1\right)}{\sum_{X \in \eta} \mathrm{sig}\left(A_N, X\right)} \\ & + \cdots + \frac{1}{2(N-1)} \times \frac{\mathrm{sig}\left(A_{i-2}, A_{i-1}\right)}{\sum_{X \in \eta} \mathrm{sig}\left(X, A_{i-1}\right)},\end{aligned} \tag{2}$$

where η is the node set of the process model, N is the size of the node set η, sig: $\eta \times \eta \longrightarrow R_0^+$ is the priority relationship assigned to each pair of nodes $A_i, A_{i+1} \in \eta$, rel_i is the relative importance of the N-element chain relationship, A_i is the starting point, and A_{i-1} is the ending node. In addition, A_0 and A_N are equivalent.

Similar to the binary conflict, the situations in which the N-ary conflict relationship is generated can also be divided into the following three categories:

(1) N-ary cycle: N activities $\{A_1, A_2, \ldots, A_N\}$ form a cycle, that is, after $\{A_1, A_2, \ldots, A_N\}$ is executed in sequence, A_N can return to activity A_1 and start again. In this case, the priority relationship between these activities is allowed in the actual process and therefore needs to be preserved.

(2) Exception: the process is executed in sequence $A_1 \longrightarrow A_2 \longrightarrow A_3, \ldots, A_{N-1} \longrightarrow A_N$, but there will be exceptions to $A_N \longrightarrow A_1$ in the actual execution process. In this case, remove the exception edge in the weaker chain structure.

(3) Concurrency: there is a parallel structure in N activities $\{A_1, A_2, \ldots, A_N\}$, and the log records the possible execution order. In this case, it is necessary to delete this conflicting sorting relationship. For example, if $A_1 \longrightarrow A_2$ and A_3 are parallel structures, that is, $A_1 \longrightarrow A_2$ and A_3 can occur in any order, the log will record the possible occurrences of $A_1 \longrightarrow A_2 \longrightarrow A_3$ and $A_3 \longrightarrow A_1 \longrightarrow A_2$. In this case, you need to delete $A_3 \longrightarrow A_1$ and $A_2 \longrightarrow A_3$ that cause conflicts.

It can be seen from the above that the flight guarantee process model essentially does not have the possibility of N-ary cycles, so the causes of N-ary conflicts are exceptions and concurrent situations. The solutions are as follows.

Determine the offset between each chain relationship and the chain relationship with the greatest relative importance, as shown in formula (3).

$$S_m\left(A_1, A_2, \ldots, A_N\right) = \left|\max \mathrm{rel}_i - \mathrm{rel}_m\right|, \quad m \in N. \tag{3}$$

On this basis, determine the chain structure with the largest offset value, $\max S_m$, and find the edge with the least relative importance on this chain structure and delete it, namely, $\min \mathrm{rel}\left(A_i, A_{i+1}\right)$.

The calculation of $\min \mathrm{rel}\left(A_i, A_{i+1}\right)$ is shown in formula (1). If the offset values of these chain structures are similar, it means that there is a less important parallel structure between the chains. This paper removes the edges that are different between the chain structures and the corresponding edges in the N-ary conflict relationship.

Unlike binary conflicts and N-ary conflicts, there is no concurrency in unary conflicts, but only self-circulation or exceptions. This situation can be resolved by creating a virtual node, removing the unary conflict, and introducing it into the fuzzy mining algorithm for edge filtering. As shown in Figure 3, after the loop is released, the priority relationship of virtual nodes B_1 and $B \longrightarrow B\_1$ can be obtained.

3.2. Local Causal Structure Construction. The second part of the framework focuses on building the MAG MB of the target variable based on the SLL algorithm under the assumption of relaxing causality sufficiency. Different from the MB of DAG, MB of MAG includes the area set and the node set related to the area set in addition to the parent-child set and spouse set. The area set is defined as follows:

Definition 7 (district Sset [23]). The district set of a target variable T in an MAG, denoted as dis(T), is a set of variables in which $\forall V_i \in \mathrm{dis}(T)$, and the path from V_i to T only contains bidirectional edges.

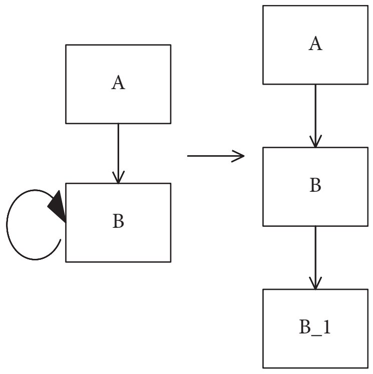

Figure 3: Conflict resolution in unary relationships.

On this basis, the article [23] proposed a method for determining whether a variable belongs to the target variable district set. Suppose $V_j \in \mathrm{adj}(T)$, $V_i \in \mathrm{adj}(V_j)$, $V_k \in \mathrm{adj}(V_i)$, if two triples $\{T, V_j, V_i\}$, $\{T, V_j, V_i\}$ satisfy: $V_i \perp\!\!\!\perp T | \mathrm{sepset}(T, V_i)$, $V_i \perp\!\!\!\perp\!\!\!\!\!/\, T | \{\mathrm{sepset}(T, V_i) \cup V_j\}$, $V_k \perp\!\!\!\perp V_j | \mathrm{sepset}(V_j, V_k)$, $V_k \perp\!\!\!\perp\!\!\!\!\!/\, V_j | \{\mathrm{sepset}(V_j, V_k) \cup V_i\}$, then $V_i \in \mathrm{dis}(V_j)$.

The above theorem shows that the variable in sp(T) is a candidate variable for the district set variable in adj(T). From this, we can further conclude whether there is a bidirectional edge between the target variable T and the variable V, as shown below.

Corollary 1. *If the triple $\{T, V_j, V_i\}$ satisfies $V_i \perp\!\!\!\perp T | sepset(T, V_i)$, $V_i \perp\!\!\!\perp\!\!\!\!\!/\, T | \{sepset(T, V_i) \cup V_j\}$ and $adj(T)/V_j$ contain variable V_m such that triple $\{V_m, T, V_j\}$ satisfies $V_j \perp\!\!\!\perp V_m | sepset(V_m, V_j)$, $V_j \perp\!\!\!\perp\!\!\!\!\!/\, V_m | \{sepset(V_m, V_j) \cup T\}$, then V_j and target variable T have bidirectional edges, that is, if there is $V_m \in adj(T)/V_j$ such that $V_m \in sp(V_j)$ in the neighborhood set of target variable T in v-structure $\{T, V_j, V_i\}$, then $V_j \in dis(T)$.*

Proof. Suppose the triplet $\{T, V_j, V_i\}$ satisfies $V_i \perp\!\!\!\perp T | \mathrm{sepset}(T, V_i)$, $V_i \perp\!\!\!\perp\!\!\!\!\!/\, T | \{\mathrm{sepset}(T, V_i))V_j\}$, and the variable V_m exists in $\mathrm{adj}(T)/V_j$ so that the triplet $\{V_m, T, V_j\}$ satisfies $V_i \perp\!\!\!\perp V_m | \mathrm{sepset}(V_m, V_j)$ and $V_i \perp\!\!\!\perp\!\!\!\!\!/\, V_m | \{\mathrm{sepset}(V_m, V_j) \cup T\}$, $V_j \notin \mathrm{dis}(T)$. If $V_j \notin \mathrm{dis}(T)$, there are no bidirectional edges between T and V_i. There are only directed edges between T and V_i, or they are independent of each other. According to the condition that the triple $\{T, V_j, V_i\}$ satisfies $V_i \perp\!\!\!\perp T | \mathrm{sepset}(T, V_i)$ and $V_i \perp\!\!\!\perp\!\!\!\!\!/\, T | \{\mathrm{sepset}(T, V_i) \cup V_j\}$, it can be seen that there is a V-structure in the triple $\{T, V_j, V_i\}$, and V_j is the collision node, and $T \longrightarrow V_j$. According to the condition that triple $\{V_m, T, V_j\}$ satisfies $V_j \perp\!\!\!\perp V_m | \mathrm{sepset}(V_m, V_j)$ and $V_j \perp\!\!\!\perp\!\!\!\!\!/\, V_m | \{\mathrm{sepset}(V_m, V_j) \cup T\}$, triple$\{V_m, T, V_j\}$ is also a V-structure, the collision node is T, and $T \longleftarrow V_j$. Therefore, the condition does not match the hypothesis, and the hypothesis does not hold. So Corollary 1 is established.

Through the above deduction, the district set adjacent to the target variable T can be judged. MAG MB includes the parents pa(T) of T, the children ch(T) of T, the spouses sp(T) of T, and the district set dis(T) of T, union of parents of each variable V_i with the district set of T, that is, $U_{i=1}^{|\mathrm{dis}(T)|}\mathrm{pa}(V_i)$, denoted as pa(dis($T$)), union of the district set each child V_i of T, that is, $U_{i=1}^{|\mathrm{ch}(T)|}\mathrm{dis}(V_i)$, denoted as dis(ch($T$)), union of parents of each variable V_i within dis(ch(T)), that is, $U_{i=1}^{|U_{j=1}^{ch(T)}\mathrm{dis}(V_i)|}\mathrm{pa}(V_i)$.

Compared with the MB discovery algorithm that uses the independence test to find the target variable T, the score-based MB discovery algorithm relies on certain scoring criteria to learn the most suitable network structure for the data sample. It has the following characteristics:

Definition 8 (local score consistency [28]). Let D be a set of data consisting of i.i.d samples from some distribution P. Let G be any BN structure and G' be the same structure as G. but with an edge from a node T to a node X. Let Pa_X^G be the parent set of X in G. A score criterion s is locally consistent if, as the size of the data D goes to infinity, the following two properties hold true:

(1) If $X \perp\!\!\!\perp\!\!\!\!\!/\, T | \mathrm{Pa}_X^G$, then $s(G, D) < s(G', D)$

(2) If $X \perp\!\!\!\perp T | \mathrm{Pa}_X^G$, then $s(G, D) > s(G', D)$

Intuitively speaking, adding an arc can eliminate independent constraints that do not exist in the data generation distribution, thereby increasing the score. Adding an arc cannot eliminate such constraints and reduce the score. Therefore, the scoring function can replace constraints to construct a causal structure to some extent. For MAG structure learning, the existing DAG scoring function cannot be directly applied to MAG. M^{3}C [29] and GSMAG [21] algorithms proposed a scoring function suitable for MAG, based on residual iterative conditional fitting to obtain the maximum likelihood estimation of a given MAG parameter. However, the M3HC and GSMAG algorithms make new assumptions on the data, which are not general. From Corollary 1, we can get the method for judging bidirectional edges. The core of the idea is to judge whether the spouse node of the target node T and the spouse node of the neighboring node V_j of T are in the other party's neighboring node. Therefore, this paper proposes a topology-based method to find the target node's neighborhood set and spouse set using the SLL [30] method. On this basis, use Corollary 1 to determine the bidirectional edge of the target node and the neighboring nodes. In this way, the neighboring district nodes of the target node are found. Finally, the complete district set of the target variable is obtained by searching the neighboring district nodes of district nodes. The algorithm is shown as follows (Algorithms 1 and 2).

The first step of the SMMB algorithm is to search for the parent and child sets of the target node based on the SLL algorithm, and the fourth to fifth steps call the FINDSPOUSES of the SLL algorithm to find the spouse node of the target node and its child nodes. The idea is as follows: the FINDNEIGHBORS algorithm is divided into two stages. The first stage searches for the potential neighbor nodes of the target variable and puts the nodes except the target node one

```
     Input: data $D$ on node set $N$ and a target node $t \in N$.
     Output: MB($t$)
(1)    $H^*(t)$, ch(t), pa(t) = FINDNEIGHBORS(D, t)
(2)    dis1, dis2, MB(t), dis = $\varnothing$
(3)    $\mathrm{MB}(t) = \mathrm{MB}(t) \cup H^*(t)$
(4)    For $q \in \mathrm{ch}(t) \cup t$ do
(5)      $S^*(q) = \mathrm{FINDSPOUSES}(D, q, H^*(q), H^*(v))$
(6)      $\mathrm{dis}^*(q) = \mathrm{FINDDIS}(D, q)$
(7)      $\mathrm{dis2} = \mathrm{dis} \cup \mathrm{dis}^*(q)$
(8)      numbers = $|\mathrm{dis2}|$
(9)      if numbers >0 then
(10)       While numbers do not change do
(11)         for $n \in$ dis do
(12)           dis1 = dis2
(13)           $\mathrm{dis}^*(n) = \mathrm{FINDDIS}(D, n)$
(14)           $\mathrm{dis2} = \mathrm{dis} \cup \mathrm{dis}^*(n)$
(15)           numbers = $|\mathrm{dis2}|$
(16)         end for
(17)     end if
(18)     for $m \in$ dis2 do
(19)       $H^*(m)\mathrm{ch}^*(m)$, $\mathrm{pa}^*(m) = \mathrm{FINDNEIGHBORS}(D, m)$
(20)       $\mathrm{pa}(\mathrm{dis}) = \mathrm{pa}(m) \cup \mathrm{pa}(\mathrm{dis})$
(21)     end for
(22)     $\mathrm{MB}(t) = \mathrm{MB}(t) \cup \mathrm{dis} \cup \mathrm{pa}(\mathrm{dis})$
(23)   end for
```

Algorithm 1: The SMMB algorithm.

```
     Input: Data D on node set N, a target node $t \in N$.
     Output: MB($t$)
(1)    $H^*(t)$, ch($t$), pa($t$) = FINDNEIGHBORS($D$, $t$)
(2)    $S^*(t) = \mathrm{FINDSPOUSES}(D, t, H^*(t), H^*(v))$
(3)    $\mathrm{dis}(t) = \varnothing$
(4)    numbers 1 = $|H^*(t)|$
(5)    if numbers1 $\geq 2$ then
(6)      for $n \in H^*(t)$ do
(7)        $H^*(n) = \mathrm{FINDNEIGHBORS}(D, n)$
(8)        for $m \in H^*(n) \backslash t$do
(9)          if $m \in S^*(t)$ then
(10)           $S^*(n) = \mathrm{FINDSPOUSES}(D, n, H^*(n), H^*(v))$
(11)           for $m1 \in S^*(n)$ do
(12)             if $m1 \in H^*(t) \backslash n$then
(13)               $\mathrm{dis}(t) = \mathrm{dis}(t) \cup n$
(14)             end if
(15)           end for
(16)         end if
(17)       end for
(18)     end for
(19)   end if
(20)   return dis(t)
```

Algorithm 2: The FINDDIS algorithm.

by one into the set Z where only the target variable exists and calls the subroutine to learn Z. Put the learned potential neighbor nodes into Z to update Z to complete the search for the set of potential neighbors. The subroutine can use a dynamic programming algorithm or other precise algorithms. In this article, the commonly used precision algorithm, the GES [31] algorithm, is used as a subroutine, and its scoring function is as follows:

$$\text{Score}_{\text{BIC}} = \sum_{i=1}^{n}\sum_{j=1}^{q_i}\sum_{k=1}^{r_i} m_{ijk}\log_2\frac{m_{ijk}}{m_{ij*}} - \sum_{i=1}^{n}\frac{q_i(r_i-1)}{2}\log_2 m. \tag{4}$$

The scoring form of the local structure formed by the target variable and its parent node is as follows:

$$\text{BIC}((X_i,\pi(X_i))|D) = \sum_{j=1}^{q_i}\sum_{k=1}^{r_i} m_{ijk}\log_2\frac{m_{ijk}}{m_{ij*}} - \frac{q_i(r_i-1)}{2}\log_2 m,$$

$$\text{Score}_{\text{BIC}} = \sum_{i=1}^{n}\text{BIC}((X_i,\pi(X_i))|D), \tag{5}$$

where m represents the number of samples, m_{ijk} represents the number of samples that meet $X_i = k$, $\pi(X_i) = j$ in the data D, X_i is the node variable in the network, and $\pi(X_i)$ represents the set of parent variables of node X_i. The node variable X_i can take a discrete value or a continuous value, $k = 1,\ldots,r_i$, r_i represents the state value of the child node, $j = 1,\ldots,q_i$, q_i represents the state value of the parent node, $m_{ij*} = \sum_{k=1}^{r_i} m_{ijk}$, the network node variable X and the data D correspond to the set Z and D_Z in the algorithm. The transformation of Z and D_Z is iterative. Change the scoring function of GES and the set of variables it can search by updating Z and D_Z. In the second stage, after searching for potential neighbor nodes, the symmetry correction is performed to remove false-positive nodes in the potential neighbor nodes to obtain the target variable's real neighbor nodes. FINDSPOUSES is similar to FINDNEIGHBORS. It searches for the target variable's potential mate set by calling a subroutine and then finds the true mate set of the target variable by forcing symmetry constraints.

In the sixth step, the SMMB algorithm calls the algorithm FINDDIS to search for the target variable's neighboring area nodes. According to Corollary 1, to determine whether a node belongs to the target node's area set, first find the spouse set of this node and the spouse set of the target node. The algorithm FINDDIS first searches the neighborhood set and spouse set of the target variable in steps 1–4, traverses the neighborhood of the target node in steps 5–7 to search for the neighborhood node of its neighborhood node, and steps 8–9 find the variables belonging to the spouse node of T in the neighboring nodes and determine the collision node n of the V-structure formed by this, steps 10–13 find the spouse node of the colliding node and determine whether it is in the neighborhood of the target node. If it is, then node n belongs to the area node of the target node.

From steps 10–16, based on searching the parent sets of the target node, pa(T), the child sets of the target node, ch(T), the spouse sets of the target node, sp(T), and the neighboring district sets of the target node, dis(T), the FINDDIS algorithm is continuously iterated to search the target node, T, and the district sets of the target variable child sets, dis(ch(T)). Steps 18–21 through the SLL algorithm to search for the Union of parents of each variable within the district set of the target node T, pa(dis(T)), and Union of parents of each variable within dis(ch(T), pa(dis(ch(T)).

In the first stage, the FINDNEIGHBORS algorithm and the FINDSPOUSES algorithm use the while loop to iterate the subroutine to search for the potential parent and child set and potential mate set of the target node. The while loop is executed at most n-1 times. The while loop is executed at most n-1 times, and the called GES algorithm determines the time and space required for each cycle. On the node set Z, the GES algorithm runs in time $O(|Z|2^{|Z|-1})$, and the computing space is $O(2^{|Z|})$. Therefore, in the worst case $|Z| = O(n)$, the time requirement of the FINDNEIGHBORS algorithm and the FINDSPOUSES algorithm in the first stage is $O(n^2 2^{n-1})$, and the space requirement is $O(n2^n)$. In the second stage of the FINDNEIGHBORS algorithm and the FINDSPOUSES algorithm, the first stage of the algorithm will be called, and it will be called up to n times. The total time of the FINDNEIGHBORS algorithm and the FINDSPOUSES algorithm is at most $O(n^3 2^{n-1})$. In the FINDDIS algorithm, FINDSPOUSES and FINDNEIGHBORS need to be called, and the number of calls is up to $(n-1)^2$ times, and the running time of the FINDDIS algorithm is up to $O((n-1)^2 n^3 2^{n-1})$. However, in practice, the network is usually relatively sparse, and the running time is significantly lower than the worst-case running time. The algorithm SMMB loops at most $(n-1)^2$ times and continuously calls the FINDDIS algorithm, the FINDNEIGHBORS algorithm, and the FINDSPOUSES algorithm in the loop. The algorithm SMMB runs at most $O(((n-1)^3+1)n^4 2^{n-1})$.

4. Experiment

In order to evaluate the quality of the method proposed in this paper, in Section 4.1, this paper uses the benchmark Bayesian network test data set alarm data set to test the method proposed in Section 3.2, and uses the evaluation index F-measure to evaluate the method proposed in Section 3.2 to prove its superiority to the common RFCI algorithm, M3B algorithm, and GFCI algorithm. Section 4.2 uses the flight guarantee data of China Xining Airport in July 2018 and the actual flight guarantee data at the airport to generate an airport flight guarantee process model, calculate the running time of each link as the performance index of each link, and extract the direction of the node edge in the process model. The SMMB algorithm is used to construct a local causal model of performance indicators and flight departure delay and adjust the direction of the one-way edge in the causal model according to the direction between the extracted process model nodes. Finally, use the MMHC algorithm to construct a local causal model of flight delays as a benchmark model and compare and analyze the causal model constructed by SMMB. The local causal model constructed by the MMHC algorithm will also be adjusted according to the direction of the edges between nodes in the process model.

4.1. Causal Discovery Algorithm Testing. The experimental test data source is the ALARM network, which contains 37 nodes and 46 edges. The network is a sparse network, which is considered a default standard for measuring the causal network construction program's level, and many algorithms and various programs have verified these data, and there is a standard network structure for comparison and reference. In order to test the performance of the above algorithm, this article uses ALARM data to randomly generate three sets of data. The first set of data includes 5 data sets of 2500 data instances, the second set of data includes 5 data sets of 5000 data instances, and the third set of data includes 5 data sets of 10,000 data instances. Then, hide some common causes in the generated data set, and treat these hidden variables as potential common causes. Specific steps are as follows:

(1) Do not hide any variables as latent variables, and this paper mines the local causal network of variable VTUB.

(2) The hidden variable INT is used as a latent variable, and this paper mines the local causal network of VTUB, where INT is the potential common cause of the variable SHNT, the variable VLNG, and the variable PRSS.

(3) Hidden variables INT and PMB are used as latent variables, and this paper mines the local causal network of VTUB. In this network, INT is the potential common cause of variable SHNT, variable VLNG, and variable PRSS, and variable PMB is the potential common cause of variable PAP and variable SHNT.

Use the dataset generated by the above steps to compare the SMMB algorithm with RFCI, M3B, and GFCI algorithms, respectively. The RFCI and M3B algorithms are both constraint-based MAG discovery algorithms. The GFCI algorithm is a global hybrid search algorithm that combines the score-based heuristic search algorithm FGS algorithm with the FCI algorithm. There are three types of conditional independence tests, G^2 test for discrete variables, Fishers$'$ Z test for continuous variables with linear relations with additive Gaussian errors, and kernel-based test for continuous variables with nonlinearity and non-Gaussian noise.

This article is the same as the literature [7]. Both RFCI and M3B algorithms are tested, and the significance level of the test is set to 0.05. The test index used is *F*-measure.

F-measure: *F*-measure combines two indicators of prediction accuracy and recall and is defined as

$$F - \text{measure} = \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}. \tag{6}$$

Precision: the prediction accuracy rate refers to the percentage of correctly predicted MNC to the total number of predicted MNI. It is used to evaluate the number of false positives in the output of the algorithm.

$$\text{precision} = \frac{\text{MNC}}{\text{MNI}}. \tag{7}$$

Recall: the recall rate refers to the ratio of the number of correctly predicted MNC to the total number. It is used to evaluate the exact number in the output of the algorithm.

$$\text{recall} = \frac{\text{MNC}}{\text{MNC} + \text{MNF}}. \tag{8}$$

MNF is the number of test samples that are not correctly identified. The index guarantee of *F*-measure combines two indicators of accuracy rate and recall rate. For the accuracy rate, recall rate, and *F*1 value obtained by each algorithm, this article takes the average of each group of 5 data sets. The comparison between SMMB algorithm and RFCI and M3B algorithms in accuracy, recall rate, and *F*1 value is shown in Table 1. Figures 4–6 are comparison diagrams of SMMB, RFCI, M3B, and GFCI algorithms in terms of accuracy, recall rate, and F1. As shown in Tables 1–3, the accuracy, recall rate, and F1 value vary with sample size.

From the accuracy comparison chart, we can clearly see that the SMMB algorithm and the M3B algorithm are relatively close in accuracy, roughly between 0.6 and 0.7, and both are better than the RFCI algorithm. In comparing the recall rate, the RFCI algorithm is much higher than the SMMB algorithm and the M3B algorithm, even up to 1. It can be seen that the RFCI algorithm contains more redundant nodes than the causal network discovered by the M3B and SMMB algorithms. Compared with the M3B algorithm, the SMMB algorithm has a certain degree of competition in accuracy, but the SMMB algorithm has a higher recall rate than the M3B algorithm. As a result, in terms of comprehensive evaluation index *F*1, the *F*1 value of the SMMB algorithm is better than that of the M3B algorithm and the RFCI algorithm. Compared with the GFCI hybrid heuristic algorithm, the SMMB algorithm has certain advantages in accuracy, recall, and *F*1, but its advantages are not obvious. The *F*1 value of the SMMB algorithm is only about 0.1 higher than the GFC algorithm.

SMMB algorithm, RFCI algorithm, and M3B algorithm are algorithms for constructing causal networks based on topology, and all need to find adjacent variables of a given target variable first. For the RFCI algorithm, the key is to find the correct graph skeleton from the data set. For the SMMB algorithm and M3B algorithm, the key is to find the neighboring nodes of the target variable. The RFCI algorithm uses the PC-stable algorithm to find the network skeleton, while the M3B algorithm uses the AdjV algorithm to find the neighboring nodes of the target variable. SMMB algorithm uses the score-based SLL algorithm framework when looking for the target node's parent-child set and spouse set. Compared with the constraint-based method such as the AdjV algorithm and the PC-stable algorithm, this algorithm searches the target node's neighborhood set. It has more advantages. The GFCI algorithm is a hybrid search algorithm, and its search for the target node's domain set is based on the FGS heuristic algorithm with higher accuracy. Therefore, the SMMB algorithm is more competitive than the GFCI algorithm. The advantages are not obvious. Compared with the GFCI algorithm, the superiority of the SMMB algorithm is more reflected in the fact that the SMMB algorithm is a local causal discovery algorithm, which takes a shorter time to build a causal network and avoids the occurrence of the Curse of Dimensionality. Therefore, based on the above reasons, we use the SMMB algorithm to discover the cause and effect of the flight guarantee process.

Table 1: When there is no latent variable, SMMB algorithm compares with RFCI, M3B, and GFCI algorithms in accuracy, recall rate, and $F1$ value.

Without latent variables (none)	2500 samples	5000 samples	10000 samples
		Precision	
SMMB	0.765	0.65	0.6842858
RFCI	0.4217364	0.222867	0.2336904
M3B	0.5733334	0.633333	0.63
GFCI	0.5	0.4444444	0.4166667
		Recall	
SMMB	0.6333334	0.6533334	0.6333334
RFCI	0.770909	1	1
M3B	0.3	0.333333	0.3666666
GFCI	0.6666667	0.6666667	0.8333333
		$F1$	
SMMB	0.6761904	0.607619	0.6482718
RFCI	0.4199646	0.3586522	0.3718414
M3B	0.3919192	0.435556	0.459394
GFCI	0.5714286	0.5333333	0.5555556

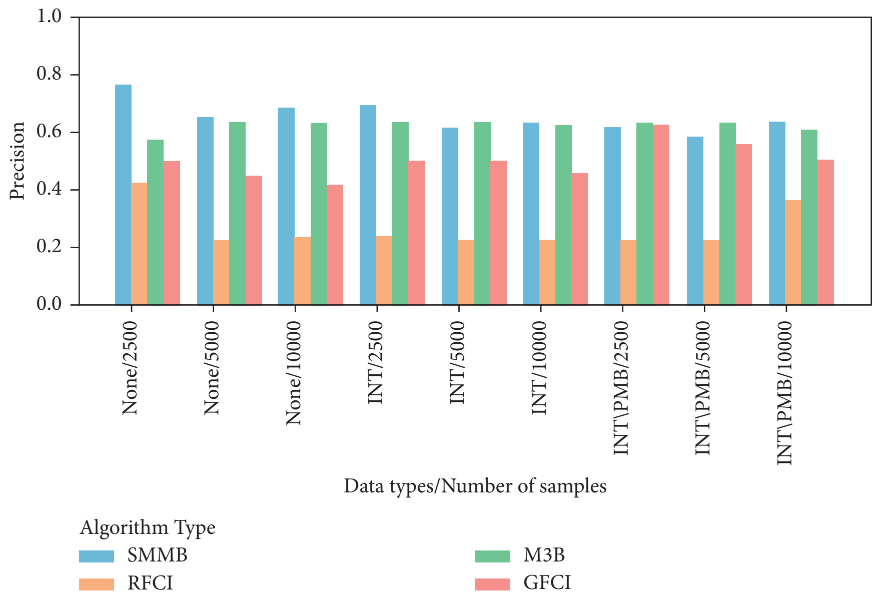

Figure 4: Comparison of SMMB, RFCI, M3B, and GFCI in precision.

4.2. Cause and Effect of Flight Service Process. The ProM used in this paper is used as a tool for mining the flight service process, using the flight guarantee data of Xining Airport in China in July 2018 for a case study. There are 122839 pieces of these data, which record the case, type, activity, resource, timestamp, and other information of Xining Airport from July 1st to July 31st. The data sample is shown in Table 4.

The fuzzy process mining algorithm plug-in in the ProM tool performs process mining on the flight service event log. The retention threshold and the ratio threshold are set to 0.27 and 0.35, respectively, to obtain the flight service process model. As mentioned in Section 3.1, the process model is optimized for the N-ary conflict and the unary conflict resolution. The optimized model is shown in Figure 7. For the node's evaluation index, this paper calculates the timestamp difference between the previous node and the next node as the duration of the previous node to evaluate the service operation efficiency of each node. For the link structure with parallel relationship, the link structure's duration is used as the basis for competition, and the longer part is selected as the overall duration of the parallel structure. In addition, the departure time of the node's front station and the node's own station is used as the evaluation index. By looking for the causal relationship between the flight launch delay time and the evaluation indicators of each flight guarantee node, we can analyze which nodes of the flight guarantee cause the final delay. Import the obtained evaluation index of each flight node into the SMMB algorithm, and the target node is flight delay. The resulting causal network of flight guarantee is shown in Figure 8. Figure 9 shows the local causal model of flight delays obtained after importing the evaluation indicators of each node of the flight

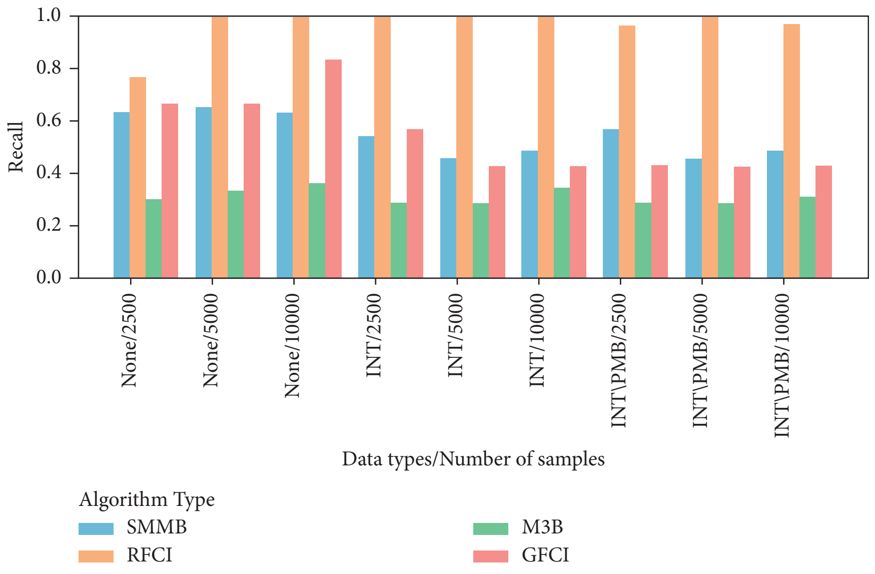

Figure 5: Comparison of SMMB, RFCI, M3B, and GFCI in recall.

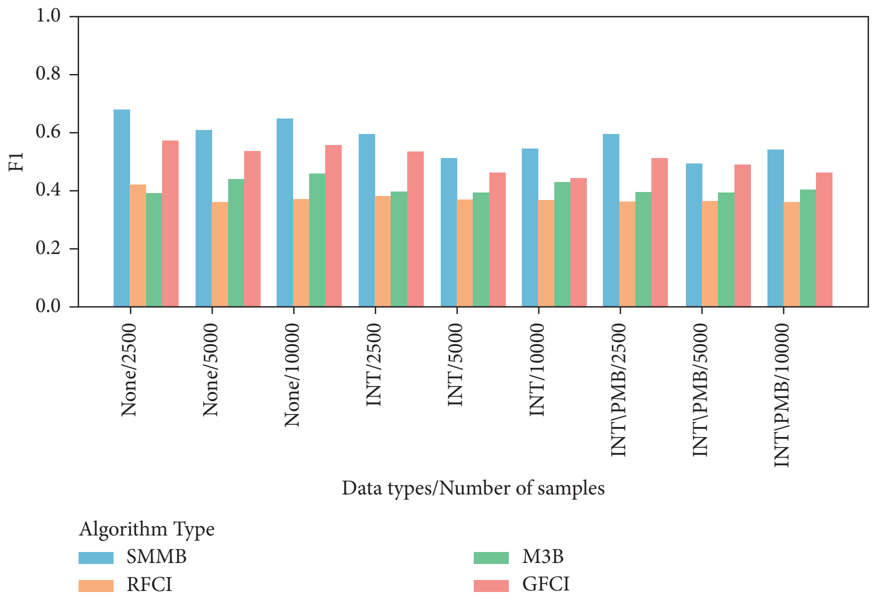

Figure 6: Comparison of SMMB, RFCI, M3B, and GFCI in $F1$.

guarantee into the MMHC algorithm. This is extracted from the complete causal model. The meaning of each node letter in the figure is shown in Table 5.

It can be seen from Figure 9 that CHECK-INSTART_ALANDINTIME, REGE-AR_REALTAKEOFF, TRACTORBIND_TRACTORSTART, MANIFESTCONV EY-_CLOSECABINGATE, CLEANPERSONARR_CLEAN START, REARCABINTRA-NSPORTCAREND_TRACTO RBIND, DSHUTTLEARR_BOARDINGSTART, TAK-EO FFATTHEFRONTSTATION_PUSHPERSONARR, PAS-SENGERU-PPERSONA-RR_DSHUTTLEARR, and DELIV ERYPERSONARR_REGEAR are considered to be the main causes of flight delays. However, in Figure 8, there are bi-directional edges between CHECKINSTART_ALAND INTIME, REGEAR_REALTAKEOFF, TRACT-ORBIND_-TRACTORSTART, MANIFESTCONVEY_CLOSECABIN GATE, and flight delays, which means that there are un-observable latent variables that affect the two sides of the bidirectional edges connection. This is also the advantage of the MAG graph to represent the causal model. The MAG graph can represent the existence of latent variables through bidirectional edges, avoiding the influence of confounding effects caused by latent variables. The causal network dia-gram discovered by the MMHC algorithm does not consider the existence of latent variables, leading to the mistaken belief that CHECKINSTART_ALANDINTIME, REGE AR_REALTAKEOF-F, TRACTORBIND_TRACTORSTA RT, and MANIFESTCONVEY_CLOSECABIN-GATE are

Table 2: When INT is used as a latent variable, SMMB algorithm is compared with RFCI, M3B, and GFCI algorithms in accuracy, recall rate, and $F1$ value.

INT as latent variables (INT)	2500 samples	5000 samples	10000 samples
		Precision	
SMMB	0.692619	0.6166666	0.634286
RFCI	0.2378153	0.2260802	0.225484
M3B	0.6333334	0.6333334	0.625
GFCI	0.5	0.5	0.454545
		Recall	
SMMB	0.5428572	0.4571428	0.485714
RFCI	1	1	1
M3B	0.2857143	0.2857142	0.342857
GFCI	0.5714286	0.4285714	0.428571
		$F1$	
SMMB	0.590696	0.5111422	0.539134
RFCI	0.3835748	0.3687198	0.367719
M3B	0.3927272	0.3927272	0.429091
GFCI	0.5333333	0.4615385	0.441176

Table 3: When INT and PMB are used as latent variables, SMMB algorithm is compared with RFCI, M3B, and GFCI algorithms in accuracy, recall rate, and $F1$ value.

INT and PMB as latent variables (INT\PMB)	2500 samples	5000 samples	10000 samples
		Precision	
SMMB	0.6157738	0.5833334	0.6342858
RFCI	0.2232481	0.223421	0.219775
M3B	0.6333334	0.6333334	0.6071428
GFCI	0.625	0.5555556	0.5
		Recall	
SMMB	0.5714285	0.4571428	0.4857142
RFCI	0.9642858	1	0.9714286
M3B	0.2857143	0.2857142	0.3142858
GFCI	0.4285714	0.4285714	0.4285714
		$F1$	
SMMB	0.58837	0.494732	0.5391342
RFCI	0.3625138	0.3651232	0.3583038
M3B	0.3927272	0.3927272	0.4062338
GFCI	0.5084746	0.483871	0.4615385

Table 4: Example log of flight service event of Xining Airport in July 2018.

Case	Type	Activity	Resource	Timestamp	Lifecycle: transition
H	Task	DROPOFFEND	h	2018-07-01T11 : 54 : 27.000	Complete
H	Task	DROPOFFCHECK	h	2018-07-01T11 : 55 : 19.000	Complete
H	Task	LUGLOADPERSONARR	h	2018-07-01T11 : 55 : 26.000	Complete

the direct causes of flight delays. There is no causal relationship in nature. Therefore, improving the four links of CHECKINSTART_ALANDINTIME, REG-EAR_REAL TAKEOFF, TRACTORBIND_TRACTORSTART, and MANIFESTCON-VEY_CLOSECABINGATE in airport flight guarantee cannot effectively improve airport flight delays. Figures 8 and 9 all believe that CLEANPERSO-NARR_CLEANSTART, REARCABINTRANSPORTCARE ND_TR-ACTORBIND, DSHUTTLEARR_BOARDINGST ART, TAKEOFFATTHEFRON-TSTATION_PUSHPERSO NARR, and PASSENGERUPPERSONARR_DSHUTT-LE ARR are the direct causes of flight delays. Therefore, if the airport wants to change the current situation of flight delays, it can start by improving the interval time of the five links of CLEANPERSONARR_CLEANSTART, REARCABINTRA NSPORTC-AREND_TRACTORBIND, DSHUTTLEARR_ BOARDINGSTART, TAKEOFFAT-THEFRONTSTATI ON_PUSHPERSONARR, and PASSENGERUPPERSO NAR-RDSHUTTLEARR, optimize the efficiency of the cleaning personnel and release personnel, and reasonably plan the driving route of tractors and shuttles, to improve the status of flight delays at the airport.

From the above, we can see that by starting from the flight service event log, the actual process model of flight service is

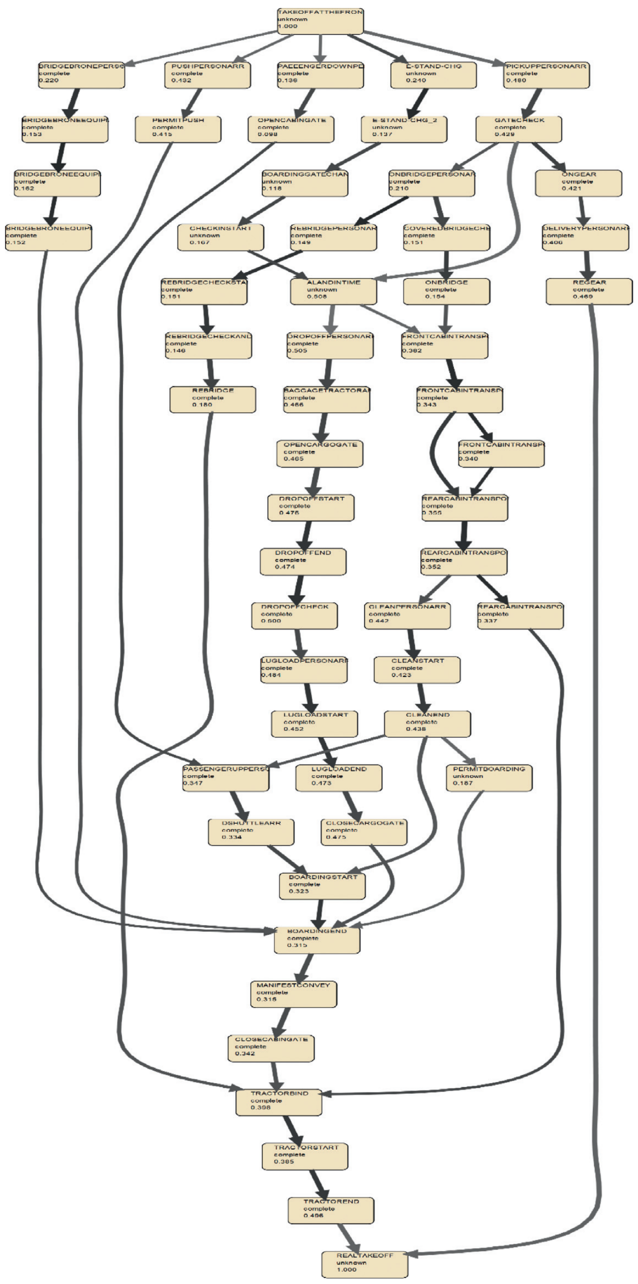

Figure 7: Process model of flight service.

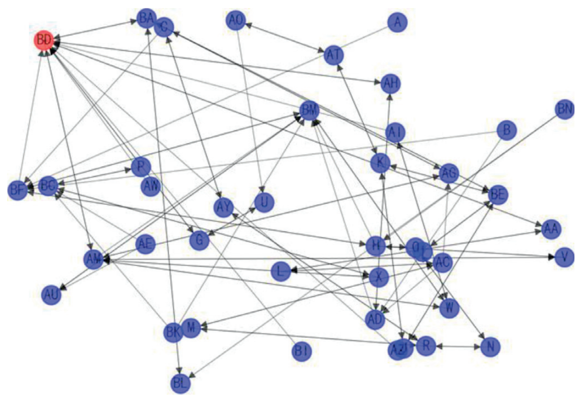

Figure 8: The cause and effect diagram of the partial process of SMMB flight guarantee.

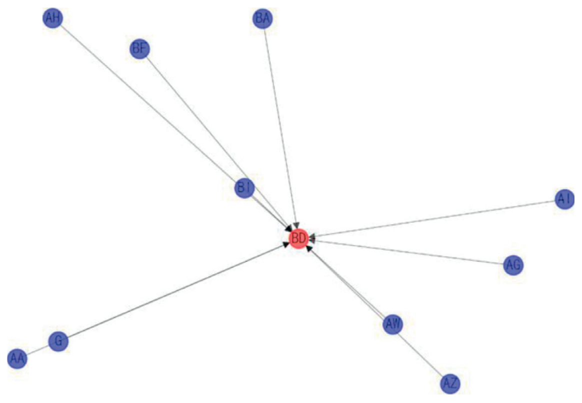

Figure 9: The causal graph of the local process of MMHC flight service.

constructed, and based on this process model, the causal study of the flight service process is carried out due to the flight service process model used in this article. It is data-driven, and it fits the actual situation. In addition to the active nodes listed in the process model, some unobservable confounding factors also affect the launch of flights, such as weather, airport layout, the operation process of flight service, and other factors. Therefore, we relax the assumption of the sufficiency of causality in Definition 2 mentioned above and use the ancestor graph to represent the causal relationship between nodes in the flight service process and reflect latent variables' existence in the form of bidirectional edges. Compared with DAG causal representation, MAG causal representation eliminates the "false" correlation between the duration of part nodes and flight delays caused by the confounding effects of latent variables. As for the direct cause of flight delays considered in the causal network, we conducted an in-depth analysis using existing data. As shown in Table 6, the number of flights served by the operators of the node is the cause of flight delays in a period of time. We can see that the scheduling system of the operators of these nodes is extremely unreasonable, and the variance of the number of flights served by each employee is extremely large, up to 133.4737. Nearly half of the employees serve far less than the average number of flights, resulting in a waste of human resources, while the other half of the employees have to fall into a high-intensity work state to maintain the operation of airport ground services. The normal work efficiency cannot be guaranteed, which leads to the extension of the operation time, which is also the root cause of the high flight delay time at the airport. We use the causal discovery method of the flight service process based on event sequence to mine the reason nodes that cause frequent flight delays at Xining Airport in China from the aspect of flight service and use the existing flight station data to conduct an in-depth analysis of each node and find the reason of nodes. Unreasonable personnel scheduling caused flight delays at the airport. Using the ACLP method proposed in this article, the airport can quickly locate the cause of airport flight delays, and use existing data to analyze

Table 5: Notes on the causal graph nodes of flight service.

A	B	C	G	H	I	J
TAKEOFFATTHEFRONTSTATION_E-STAND-CHG	TAKEOFFATTHEFRONTSTATION_PICKUPPERSONARR	TAKEOFFATTHEFRONTSTATION_PAEEENGERDOWNPERSONARR	CHECKINSTART_ALANDINTIME	ALANDINTIME_DROPOFFPERSONARR	BRIDGEBRONEPERSONARR_BRIDGEBRONEEQUIPMENTARR	BRIDGEBRONEEQUIPMENTARR_BRIDGEBRONEEQUIPMENTSTART
K	L	M	N	O	P	R
BRIDGEBRONEEQUIPMENTSTART-BRIDGEBRONEEQUIPMENTEND	DROPOFFPERSONARR_BAGGAGETRACTORARR	BAGGAGETRACTORARR_OPENCARGOGATE	OPENCARGOGATE_DROPOFFSTART	DROPOFFSTART_DROPOFFEND	DROPOFFEND_DROPOFFCHECK	LUGLOADPERSONARR_LUGLOADSTART
U	V	W	X	AA	AC	AD
PUSHPERSONARR-PERMITPUSH	PERMITBOARDING_BOARDINGEND	CLOSECARGOGATE_BOARDINGEND	BRIDGEBRONEEQUIPMENTEND-BOARDINGEND	MANIFESTCONVEY_CLOSECABINGATE	GATECHECK_ONGEAR	GATECHECK_ALANDINTIME
AE	AG	AH	AI	AM	AO	AT
GATECHECK-ONBRIDGEPERSONARR	DELIVERYPERSONARR_REGEAR	REGEAR_REALTAKEOFF	CLEANPERSONARR_CLEANSTART	BOARDINGSTART_BOARDINGEND	ONBRIDGEPERSONARR_COVEREDBRIDGECHECK	COVEREDBRIDGECHECK_ONBRIDGE
AU	AW	AY	AZ	BA	BC	BD
FRONTCABINTRANSPORTCARSTART-FRONTCABINTRANSPORTCAREND	REARCABINTRANSPORTCAREND_TRACTORBIND	PAEEENGERDOWNPERSONARR_OPENCABINGATE	DSHUTTLEARR_BOARDINGSTART	TRACTORBIND_TRACTORSTART	TRACTOREND_REALTAKEOFF	FLIGHTDELAY
BE	BF	BI	BK	BL	BM	BN
TAKEOFFATTHEFRONTSTATION-BRIDGEBRONEPERSONARR	TAKEOFFATTHEFRONTSTATION_PUSHPERSONARR	PASSENGERUPPERSONARR_DSHUTTLEARR	FRONTCABINTRANSPORTCARSTART_REARCABINTRANSPORTCARBIND	ONBRIDGE_FRONTCABINTRANSPORTCARBIND	PERMITPUSH_BOARDINGEND	ALANDINTIME_FRONTCABINTRANSPORTCARBIND

Table 6: The operation status of flight support service personnel.

	AZ	BI	AI	AW	BF
Mean	68.107	68.756	265.778	49.279	133.474
Variance	40.534	40.577	113.264	36.46	90.363
Ratio	0.536	0.536	0.222	0.512	0.474

Mean: the average number of flights served by the operating personnel; variance: the variance of the number of flights served by the operator; ratio: the proportion of the operators whose number of flights served is less than the average.

the causes from different angles, and explore the reason for flight delays at a deeper level. However, there are still some shortcomings in this article. This article only searches for flight delays from a qualitative point of view and does not involve quantitative measures. Therefore, it is impossible to further answer the specific impact of the cause nodes of the flight delay on the flight delay. When the airport side is taken to adjust the nodes for these reasons, we cannot know whether these measures have an improvement effect on flight delays and how much of an effect they have. This is also the next step to consider.

5. Conclusions and Future Work

This paper uses a new framework to automatically discover the root cause of business process performance problems. Compared with the automatic causal discovery of business process performance based on the Granger causality test proposed by some researchers, the method proposed in this paper uses the SMMB algorithm. Instead of the Granger causality test to solve the influence of potential confounding effects, the node pairs with latent variables are marked by bidirectional edges. However, this article only discovers the causal relationship between business performance indicators represented by flight delays and node operation duration and does not explore the specific causal effects of each node's duration on flight delays. That is, this problem is equivalent to the problem of estimating causal effects when covariates are lacking. In future work, we will further explore this problem by estimating the causal effect of each node on flight delays, look for nodes that have a greater impact on delays, and take appropriate measures concerning these nodes to improve the business performance of the airport.

Data Availability

The airport ground support database was purchased from Airports Council International (ACI). To have access, the researcher must get in touch with this organization and acquire the data. The alarm database comes from the bnlearn package.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This research was supported by the Sichuan Science and Technology Program and the Second Research Institute of Civil Aviation Administration of China under Grant nos. 2020YFG0050 and 2019JDTD0001.

References

[1] Civil Aviation Administration of China, *2019 Statistical Bulletin on the Development of the Civil Aviation Industry*, pp. 46-47, Civil Aviation Management, Beijing, China, 2020.

[2] Z. Xing, Z. Wei, and Q. Luo, "Flight support service process modeling method based on colored time Petri net," *Journal of Systems Engineering and Electronics*, vol. 40, no. 464, pp. 109–114, 2018.

[3] H. Jing-qi, Y. Wen-dong, and T. Xiao-wei, "Simulation for platform truck resource allocation in hub airport apron," *Journal of Wuhan University of Technology*, vol. 35, no. 12, pp. 85–91, 2013.

[4] Z. Xing and T. Yunxiao, "Flight support service time estimation of hub airport," *Journal of System Simulation*, vol. 29, no. 11, pp. 2856–2864, 2017.

[5] I. Kovynyov and R. Mikut, "Digital technologies in airport ground operations," *Netnomics: Economic Research and Electronic Networking*, vol. 20, no. 3, pp. 1–30, 2019.

[6] J. Rudnitckaia, *Process Mining. Data Science in Action*, pp. 1–11, Eindhoven University of Technology, Faculty of Information Technology, Eindhoven, Netherlands, 2015.

[7] P. Spirtes, C. Glymour, and R. Scheines, *Causation, Prediction, and Search*, The MIT Press, Cambridge, MA, USA, 2 edition, 2000.

[8] C. D. Ciccio, A. Marrella, and A. Russo, "Knowledge-Intensive processes: characteristics, requirements and analysis of contemporary approaches," *Journal on Data Semantics*, vol. 38, no. 3, pp. 29–57, 2015.

[9] J.-j. Wang, *Research on Some Key Issues of Business Process Management Based on Event Log*, Hangzhou Dianzi University, Hangzhou, China, 2015.

[10] A. Corallo, M. Lazoi, and F. Striani, "Process mining and industrial applications: a systematic literature review," *Knowledge and Process Management*, vol. 27, no. 3, pp. 225–233, 2020.

[11] W. van der Aalst, T. Weijters, and L. Maruster, "Workflow mining: discovering process models from event logs," *IEEE Transactions on Knowledge and Data Engineering*, vol. 16, no. 9, pp. 1128–1142, 2004.

[12] A. J. M. M. Weijters and J. T. S. Ribeiro, "Flexible Heuristics Miner (FHM)," in *Proceedings of 2011 IEEE Symposium on Computational Intelligence and Data Mining (CIDM)*, IEEE, Paris, France, April 2011.

[13] S. Montani, G. Leonardi, S. Quaglini, A. Cavallini, and G. Micieli, "A knowledge-intensive approach to process similarity calculation," *Expert Systems with Applications*, vol. 42, no. 9, pp. 4207–4215, 2015.

[14] M. Solé and J. Carmona, "Amending C-net discovery algorithms," in *Proceedings of the 28th Annual ACM Symposium on Applied Computing*, Coimbra, Portugal, March 2013.

[15] C. W. Günther and W. Aalst, "Fuzzy mining–adaptive process simplification based on multi-perspective metrics," in *Proceedings of International Conference on Business Process*

Management, Bpm, Springer-Verlag, Brisbane, Australia, September 2007.

[16] J. Pearl, "Probabilistic reasoning in intelligent systems: networks of plausible inference," *Artificial Intelligence*, vol. 48, no. 8, pp. 117–124, 1990.

[17] I. Tsamardinos, C. F. Aliferis, and A. R. Statnikov, "Algorithms for large scale Markov blanket discovery," in *Proceedings of FLAIRS Conference*, vol. 2, pp. 376–380, Augustine, FL, USA, May 2003.

[18] I. Tsamardinos, C. F. Aliferis, and S. Alexander, "Time and sample efficient discovery of Markov blankets and direct causal relations," in *Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Washington, DC, USA, August 2003.

[19] C. F. Aliferis, A. Statnikov, I. Tsamardinos, S. Mani, and X. D. Koutsoukos, "Local causal and Markov blanket induction for causal discovery and feature selection for classification part I: algorithms and empirical evaluation," *Journal of Machine Learning Research*, vol. 11, no. 7, pp. 171–234, 2010.

[20] C. F. Aliferis, A. Statnikov, I. Tsamardinos, S. Mani, and X. D. Koutsoukos, "Local causal and Markov blanket induction for causal discovery and feature selection for classification part II: analysis and extensions," *Journal of Machine Learning Research*, vol. 11, pp. 171–234, 2010.

[21] S. Triantafillou and I. Tsamardinos, "Score-based vs constraint-based causal learning in the presence of confounders," in *Proceedings of the Conference on Uncertainty in Artificial Intelligence (UAI)*, pp. 59–67, Jersey, NJ, USA, June 2016.

[22] T. Richardson and P. Spirtes, "Ancestral graph Markov models," *Annals of Statistics*, vol. 30, no. 4, pp. 962–1030, 2002.

[23] K. Yu, L. Liu, J. Li, and H. Chen, "Mining Markov blankets without causal sufficiency," *IEEE transactions on neural networks and learning systems*, vol. 29, no. 12, pp. 6333–6347, 2018.

[24] W. Van der Aalst, A. Adriansyah, and B. van Dongen, "Replaying history on process models for conformance checking and performance analysis," *WIREs Data Mining and Knowledge Discovery*, vol. 2, no. 2, pp. 182–192, 2012.

[25] L. S. González, F. G. Rubio, F. R. González, and M. P. Velthuis, "Measurement in business processes: a systematic review," *Business Process Management Journal*, vol. 16, no. 1, pp. 1463–7154, 2010.

[26] B. F. A. Hompes, A. Maaradji, M. La Rosa, M. Dumas, J. C. A. M. Buijs, and W. M. P. van der Aalst, "Discovering causal factors explaining business process performance variation," *Advanced Information Systems Engineering*, vol. 10253, pp. 177–192, 2017.

[27] C. W. Günther and W. M. P. Van Der Aalst, "Fuzzy mining-adaptive process simplification based on multi-perspective metrics,"vol. 4714, pp. 328–343, in *Proceedings of International Conference on Business Process Management*, vol. 4714, , Springer, Berlin, Heidelberg, September 2007.

[28] D. M. Chickering, "Optimal structure identification with greedy search," *Journal of Machine Learning Research*, vol. 3, pp. 507–554, 2002.

[29] K. Tsirlis, V. Lagani, S. Triantafillou, and I. Tsamardinos, "On scoring maximal ancestral graphs with the max-min hill climbing algorithm," *International Journal of Approximate Reasoning*, vol. 102, no. 7, pp. 74–85, 2018.

[30] T. Niinimaki and P. Parviainen, "Local structure discovery in Bayesian networks," arXiv preprint arXiv:1210.4888, 2012.

[31] T. Silander and P. Myllymaki, "A simple approach for finding the globally optimal bayesian network structure," arXiv preprint arXiv:1206.6875, 2012.