SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

March 2025, Vol. 68, Iss. 3, 132105:1–132105:15
https://doi.org/10.1007/s11432-023-4203-6

Federated local causal structure learning

Kui YU[1*†], Chen RONG[1†], Hao WANG[1], Fuyuan CAO[2] & Jiye LIANG[2]

[1]*School of Computer and Information, Hefei University of Technology, Hefei 230601, China*
[2]*School of Computer and Information Technology, Shanxi University, Taiyuan 030006, China*

Received 28 October 2023/Revised 7 April 2024/Accepted 9 August 2024/Published online 16 January 2025

Abstract Local causal structure learning (LCS) efficiently identifies a set of direct neighbors of a specified variable from observational data. Additionally, it distinguishes direct causes and direct effects of this variable without learning the entire causal structure. While many LCS algorithms have been proposed, they do not consider the data privacy-preserving problem, which has attracted extensive attention from academia and industry. To address this issue, we propose a federated local causal structure learning (FedLCS) algorithm to learn local causal structures in privacy-preserving data in a federated setting. Specifically, FedLCS introduces a layer-wise federated local skeleton learning algorithm to construct the local skeleton. Based on this skeleton, it introduces a federated local skeleton orientation algorithm and an extension-and-backtracking orientation algorithm to orient the edges. Finally, FedLCS uses a federated local extension-and-backtracking orientation algorithm to orient the remaining edges. Extensive experiments on benchmark, synthetic, and real datasets demonstrate that FedLCS can learn the local causal structure of a given variable in a federated setting.

Keywords local causal structure learning, federated learning, directed acyclic graph, privacy-preserving data, federated layer-wise strategy

Citation Yu K, Rong C, Wang H, et al. Federated local causal structure learning. Sci China Inf Sci, 2025, 68(3): 132105, https://doi.org/10.1007/s11432-023-4203-6

1 Introduction

Causal structure learning (CSL) discovers causal relationships between variables from observational data and has been widely used in economics, medicine, and many other research fields [1, 2]. For example, learning from sales data to determine whether a causal relationship exists between advertising placement and product sales is crucial for promoting better sales strategies. A causal structure is often represented by a directed acyclic graph (DAG). In such a DAG, if a directed edge points from variable X_i to variable X_j, then X_i is considered the direct cause of X_j, and X_j is the direct effect of X_i [3].

Recently, many CSL methods have been designed [4, 5]. These methods can be broadly divided into global CSL and local CSL approaches. The first type aims to learn a(n) global or entire causal structure of all variables in a dataset, such as ADL [6] and NOTEARS [7]. The second type learns the local causal structure of a target variable, specifically the direct causes (parents) and direct effects (children) of that variable, without constructing a global causal structure, such as ELCS [8] and CMB [9]. In many practical scenarios, learning a global structure is unnecessary and resource-intensive when the goal is to understand causal relationships around a given variable [10]. In machine learning, identifying the direct causes and effects of the class attribute through local causal structure learning (LCS) offers an interpretable and robust feature selection approach for classification tasks [11]. Most importantly, it is also pivotal for identifying invariant features (i.e., direct causes) across domains in domain generalization tasks [12].

Most existing CSL methods require aggregating datasets from different groups/organizations into a single dataset or sharing datasets among groups/organizations. However, preserving data privacy has recently gained significant attention owing to increasing incidents of data abuse and leakage [13, 14]. Therefore, datasets are often kept isolated across different organizations/groups to protect privacy concerns. For instance, it is challenging to collect patients' electronic medical records from different hospitals for advanced medical data analysis owing to patients' privacy preservation. Given the growing need

* Corresponding author (email: yukui@hfut.edu.cn)
† These authors contributed equally to this work.

for preserving data privacy in medicine, finance, and other applications [15–17], federated learning has emerged as a prominent research direction [18]. Federated learning adopts the paradigm of "data stays and computation moves" to ensure data privacy by exchanging model parameters among different clients (i.e., organizations or groups) for joint computations on a central server, without sharing or aggregating the original data stored in different clients [19].

Although several algorithms have been proposed to learn a global causal structure with data privacy considerations, such as NOTEARS-ADMM [20], FedDAG [21], and FedPC [22], the development of privacy-aware algorithms for LCS remains unexplored. This brings an important research issue as federated local causal structure learning is significant for real-world applications. For instance, learning the factors that cause chronic diseases from patients' electronic medical records across different hospitals-while preserving patient privacy-can provide valuable insights for public health policies and recommendations. Moreover, learning the local causal structure around the class attribute is a promising method for generalizable federated learning [23].

This study proposes a federated local causal structure learning (FedLCS) algorithm to learn a local causal structure in a federated setting, which comprises three novel subroutines: federated local skeleton learning (FLSke), federated local skeleton orientation (FLSori), and federated local extension-and-backtracking orientation (FLEori). To our knowledge, this study is the first to learn local causal structures in privacy-preserving scenarios.

First, in the FLSke subroutine, we propose a novel layer-wise local skeleton learning and sharing strategy. This layer-wise strategy enables each client to learn the skeleton independently and share the learned skeleton parameters on a server at each layer without centralizing data from each client to the server. Through this strategy, the parameters of skeleton learning at each layer can be fully exchanged to jointly learn a reliable skeleton. Second, in the FLSori subroutine, we introduce an effective strategy to deal with the consistent separation set problem for locally identifying accurate v-structures in the local skeleton learned in the FLSke subroutine. Third, based on the results of the FLSori subroutine, in the FLEori subroutine, using the FLSke subroutine, we design an FLEor strategy to orient the edges connected to a given target variable as many as possible.

We conducted extensive experiments on benchmark datasets, synthetic datasets, and a real dataset to validate the effectiveness of the FedLCS algorithm.

2 Related work

Existing LCS algorithms typically begin with a target variable. Subsequently, standard Markov blanket (MB) or parents and children (PC) learning algorithms are applied to learn the local skeleton of the variable. These algorithms recursively identify v-structures to orient the undirected edges of the local skeleton [24]. The local skeleton is extended until all undirected edges are oriented. However, these LCS algorithms have been developed for single datasets without considering data privacy.

The PCD-by-PCD (where PCD denotes parents, children, and some descendants) uses the max-min parents and children (MMPC) algorithm [25] to learn a local skeleton of a given variable and determine separation sets. These separation sets are subsequently used to identify v-structures. Finally, using the identified v-structures and the Meek-rules [26], the algorithm orients the edges as much as possible. Considering that identifying the MB is easier and faster than finding the PCD of a target variable in a large causal structure, the MB-by-MB algorithm uses various MB discovery methods, such as MMMB [27] and EAMB [28] for local skeleton learning [29]. Unlike the aforementioned algorithms, the CMB algorithm first identifies the MB set of a target variable and then orients the edges by tracking the conditional independence changes in the MB of the target [9].

To address the issue of unreliable CI tests due to the data noise or small data sample sizes, the FS-LCS algorithm [30] improves skeleton learning efficiency by using a filter feature selection method instead of performing CI tests to find PCs and separation sets. The algorithm then recursively identifies spouses to aid in edge orientations, continuing until the direct causes and direct effects of a given target variable are distinguished. The partial BN structure learning (PSL) algorithm was developed to learn a partial causal structure and does not only focus on direct causes and effects of a target variable [31].

To address the data privacy protection problem, McMahan et al. [32] introduced the federated learning concept in 2016 to address the conflicts between data privacy protection and machine learning. Federated learning, a data-sharing computing paradigm, has since become a key research area for addressing data

Table 1 Summary of notations.

Notation	Meaning
V	Variable set $\{X_1, X_2, \ldots, X_n\}$
X_i, X_j	Individual variable
T	The target variable
Z	Conditioning set
sepset(X_i, X_j)	Separation set of X_i andX_j
S_T^c	Initially local skeleton of T
$S_T^{l=1}$	Local skeleton of T learned at the $l = 1$ layer
S_T^*	Final local skeleton of T learned at the FLSke subroutine
PC_T, PC_{X_i}	PC of T and X_i respectively
CPC_T^l	Candidate PC of T learned at the l-th layer
CPC_T^*	Final PC of T learned at the FLSke subroutine
S_T^{*1}	Local skeleton of T with depth = 1
S_T^{*2}	Local skeleton of T with depth = 2
N	Number of the clients
D	Dataset $D = \{D_1, D_2, \ldots, D_N\}$
K	Number of all variables
l	Number of the layer (i.e., the size of conditioning set)

privacy concerns. Within this paradigm, entities generating datasets are known as clients, while the platform aggregating their model parameters is referred to as the server. Federated learning is divided into horizontal and vertical federated learning based on client data distribution differences. Horizontal federated learning involves clients with varying sample sizes sharing identical features, ideal for scenarios with overlapping user features but distinct users. Vertical federated learning focuses on feature union, fitting situations where users overlap, and their features are unique. Owing to the page limit, further details of federated learning and appropriate references can be found in good survey papers [14, 33, 34].

In this paper, we focus on learning local causal structures in horizontal federated learning. Several algorithms have been designed to learn a global causal structure in horizontal federated settings. For instance, NOTEARS-ADMM is based on the NOTEARS method that formulates structure learning of linear BNs as a continuous constrained optimization problem. It adopts a distributed optimization method known as the ADMM to split the constrained problem. It then uses the augmented Lagrangian method to solve the constrained minimization problem. Each client computes its parameters in each iteration based on the local dataset and sends them to the server. The server uses these local parameters to calculate the global parameters and returns them to each client to enter the next iteration. Therefore, each client exposes only its parameters and not information related to its local dataset. In contrast, FedDAG is a gradient-based learning framework, which has a two-level structure for each local model. In the first level, each client makes local gradient-based parameter updates to maximize its personalized score and then learns a federated DAG. The server selects several clients randomly, aggregates their outcomes by averaging, and then updates the clients with this result for iterative learning. The second level uses a neural network to separately approximate the causal mechanisms and personal updates from its data to accommodate the data heterogeneity. FedPC was the first algorithm that integrated the well-known PC algorithm with the federated learning framework. It employs a federated layer-wise strategy for sharing and aggregating skeletons at each layer between clients and the server and the learned adjacency matrix and separation set information are only exchanged between clients and the server.

Although several algorithms have been designed to learn a local causal structure, the development of algorithms that consider data privacy remains unexplored. Thus, this study proposes algorithms to address the local causal structure problem in the federated setting.

3 Notation and definitions

The notations frequently used in the paper are summarized in Table 1.

Definition 1 (Conditional independence)**.** Given a set Z, two variables X_i and X_j are conditionally independent if and only if $P(X_i, X_j|Z) = P(X_i|Z)P(X_j|Z)$. For example, in Figure 1(c), given that $A \to T \to C$ is a chain structure, A and C are independent conditioning on $\{T\}$.

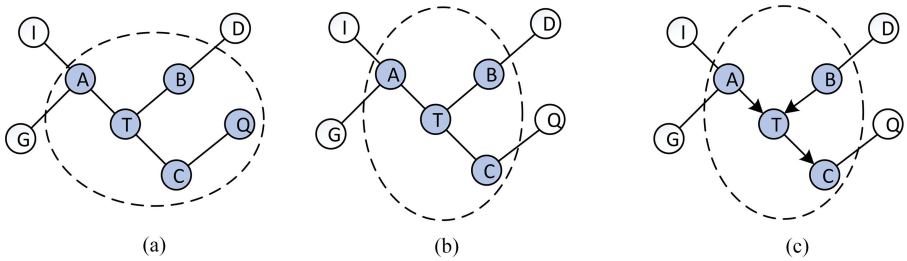


Figure 1 (Color online) (a) MB(T) = $\{A, B, C, Q\}$ and (b) PC(T) = $\{A, B, C\}$ denote the local causal skeletons of T, and (c) is the local causal structure of T.

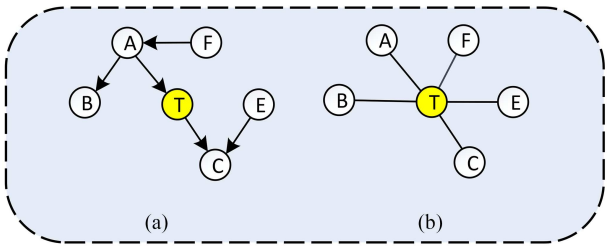


Figure 2 (Color online) (a) True structure of T, and (b) initial skeleton of T by taking A, B, C, E, F as candidate PCs.

X_i and X_j are independent conditioning on Z, denoted as $X_i \perp\!\!\!\perp X_j | Z$, while X_i and X_j are dependent conditioning on Z, denoted as $X_i \not\perp\!\!\!\perp X_j | Z$. Z is called a conditioning set and the size of a conditioning set is denoted as l.

Definition 2 (Separation set [35]). Suppose X_i and X_j are conditionally independent conditioning on a variable set Z. Then, Z represents a separation set, which is denoted as sepset(X_i, X_j). For example, in Figure 1(c), T is a separation set of A and C, denoted as sepset$(A, C) = \{T\}$.

Definition 3 (Consistent separation set). In the federated setting, X_i and X_j may have different separation sets computed across all clients that make them conditionally independent (owing to varied data quality). Among those separation sets, the one that makes X_i and X_j independent in the ground-truth causal structure is considered the consistent separation set of X_i and X_j.

Definition 4 (V-structure [35]). An undirected triple in a local skeleton such as X_i-X_k-X_j, where X_i and X_j are not directly connected, forms a v-structure if X_i and X_j are conditionally independent and X_k is not in sepset(X_i, X_j). For example, in Figure 1(c), as sepset(A, B) is an empty set, $A \to T \leftarrow B$ is a v-structure.

Definition 5 (AND rule). If X_i is a parent of X_j (i.e., $X_i \in \mathrm{PC}_{X_j}$), then X_j is a child of X_i (i.e., $X_j \in \mathrm{PC}_{X_i}$). As shown in Figure 1(c), if A is a parent of T, then T must be a child of A.

Definition 6 (Local causal skeleton). The local causal skeleton of T is referred to as the MB skeleton or PC skeleton shown respectively in Figures 1(a) and (b), which are undirected graphs consisting of T and its MB or PC variables (herein, local causal skeleton refers to the PC skeleton).

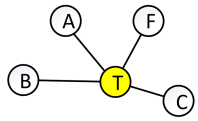
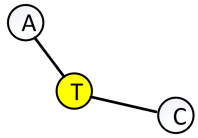
Definition 7 (Local causal structure). The local causal structure of T is a DAG only consisting of T and its PC variables, as shown in Figure 1(c).

4 Background knowledge of the PCsimple algorithm

PCsimple is a basic and traditional algorithm of local causal skeleton learning [36] in the non-federated setting. Further, several other well-established local skeleton learning algorithms have been derived from PCsimple.

It may be assumed that T is the target variable and the remaining variables in $V \setminus \{T\}$ (all variables in V excluding T) are the candidate PCs of T (i.e., $\mathrm{CPC}_T = X \setminus \{T\}$). Then, all the variables in CPC_T and the target variable T form the initial skeleton of T, denoted as S_T^c. For example, assuming that a dataset contains six variables, $\{A, B, C, E, F, T\}$ and T is the target variable, the initial skeleton of T is shown in Figure 2(b). Starting from the initial skeleton, PCsimple updates CPC_T by deleting the variables that are conditionally independent of T layer by layer with the increasing size of the conditioning set l from

Table 2 (Color online) Execution process at each layer.

l-th layer	Checking variable	CI test results	Description	CPC_T	Skeleton
$l=0$	A	$A \not\perp\!\!\!\perp T\|\phi$	Keep A	$\{A,B,C,E,F\}$	
	B	$B \not\perp\!\!\!\perp T\|\phi$	Keep B	$\{A,B,C,E,F\}$	
	C	$C \not\perp\!\!\!\perp T\|\phi$	Keep C	$\{A,B,C,E,F\}$	
	E	$E \perp\!\!\!\perp T\|\phi$	Delete E	$\{A,B,C,F\}$	
	F	$F \not\perp\!\!\!\perp T\|\phi$	Keep F	$\{A,B,C,F\}$	
$l=1$	A	$A \not\perp\!\!\!\perp T\|B$	Keep A	$\{A,B,C,F\}$	
	A	$A \not\perp\!\!\!\perp T\|C$	Keep A	$\{A,B,C,F\}$	
	A	$A \not\perp\!\!\!\perp T\|F$	Keep A	$\{A,B,C,F\}$	
	B	$B \perp\!\!\!\perp T\|A$	Delete B	$\{A,C,F\}$	
	C	$C \not\perp\!\!\!\perp T\|A$	Keep C	$\{A,C,F\}$	
	C	$C \not\perp\!\!\!\perp T\|F$	Keep C	$\{A,C,F\}$	
	F	$F \perp\!\!\!\perp T\|A$	Delete F	$\{A,C\}$	

At the $l=2$ layer, PCsimple ends since $|\mathrm{CPC}_T^1| = l$ holds.

0 until the size of CPC_T is less than l. For the true structure and initial skeleton in Figure 2, Table 2 presents an example of tracing the execution process of PCsimple for learning the local skeleton of T.

Layer 0: the value of l starts from 0. At this layer, the size of the conditioning set Z is 0 ($Z=\emptyset$). For each variable X_i in the initial skeleton S_T^c, if $X_i \perp\!\!\!\perp T|\emptyset$ holds, X_i is discarded from CPC_T and the edge between X_i and T is deleted from S_T^c accordingly. Meanwhile, $\emptyset$ is saved as the separation set of X_i and T. When all variables in S_T^c are examined at the $l=0$ layer, we obtain the set CPC_T^0 and the skeleton $S_T^{l=0}$.

Layer 1: the value of l is increased to 1. At this layer, the size of the conditioning set Z equals 1. On the basis of CPC_T^0 and $S_T^{l=0}$ obtained at the $l=0$ layer, for each variable X_i in CPC_T^0, if there exists a subset $Z \subseteq \mathrm{CPC}_T^0 \setminus \{X_i\}$ such that $|Z|=1$ and $X_i \perp\!\!\!\perp T|Z$ hold, X_i and the edge between it and T are deleted from CPC_T^0 and $S_T^{l=0}$, respectively. The conditioning set Z is saved as the separation set of X_i and T. After examining all variables in CPC_T^0, we achieve the set CPC_T^1 and the skeleton $S_T^{l=1}$ at the $l=1$ layer.

After the $l=1$ layer, the value of l is increased by 1 at each iteration. This process proceeds continuously, layer by layer, until the PC size of T at the current layer is equal to or less than l. S_T^* is saved as the final local skeleton of T.

5 Proposed FedLCS algorithm

The proposed FedLCS algorithm consists of three subroutines: FLSke, FLSori, and FLEori.

Without touching each client's original data, the FLSke subroutine learns a local skeleton of the target variable T by using the clients' datasets $\{D_1, D_2, \ldots, D_N\}$ through a novel layer-wise local skeleton learning and sharing strategy. Subsequently, the FLSori subroutine extends the learned skeleton obtained in the FLSke subroutine and then locally identifies v-structures in this extended skeleton by using a new strategy for learning effectively consistent separation sets. Finally, the FLEori subroutine orients the remaining undirected edges in the local skeleton of T. The details of the three subroutines are described in the following subsections.

5.1 FLSke

Motivated by the layer-wise concept in PCsimple, we innovatively integrated the PCsimple algorithm into the federated setting and designed the FLSke subroutine for local skeleton learning seeking to preserve data privacy. This FLSke subroutine makes all clients send their learned local skeletons of T to the server at each layer and obtain a better initial skeleton from the server by aggregating all clients' learned skeletons (the details are described in Phases 3 and 4 of Subsection 5.1), which can effectively mitigate inaccurate skeleton learning results at client sides. For example, if a client learns a low-quality skeleton of T at the l-th layer owing to limited data samples, the server improves the skeleton quality by skeleton aggregating at this layer for its $(l+1)$-layer learning, resulting in significant improvement in the accuracy of local skeleton learning.

It may be assumed that there are N clients and one server in the federated environment. Consequently, the initial skeleton S_T^c of T is an undirected graph indicating that T has an undirected edge with all remaining variables, and the initial PC set of T, $\mathrm{CPC}_T^c = V \setminus T$. The FLSke subroutine presents a novel layer-wise FLSke and sharing strategy, and it contains 4 phases at each layer, as shown in Figure 3.

Phase 1. At Phase 1, by setting the size of the conditioning to l, each client uses the local skeleton of T aggregated by the server as the initial skeleton, and it updates this local skeleton independently using its data. At the beginning of the $l = 0$ layer, at each client, the initial local skeleton and the PC set of T are S_T^c and $\mathrm{CPC}_T^c = V \setminus T$, respectively. Otherwise, at the other l ($l > 0$) layer, the initial local skeleton and the PC set of T are the skeleton S_T^{l-1} and the PC set CPC_T^{l-1} obtained by the server's aggregation at the $l-1$ layer. Specifically, at the l layer, at each client, for each variable in CPC_T^{l-1} (or CPC_T^c if $l = 0$ holds), if $\exists Z \subseteq \mathrm{CPC}_T^{l-1}$ (or CPC_T^c if $l = 0$ holds) and $|Z| = l$ such that $X_i \perp\!\!\!\perp T | Z$ holds, X_i and the edge between it and T are deleted from CPC_T^{l-1} (or CPC_T^c if $l = 0$ holds) and S_T^{l-1} (or S_T^c if $l = 0$ holds), respectively.

After all clients independently learn their local skeletons of T at the l-th layer, each client obtains an updated local skeleton S_n^l and the PC set CPC_n^l ($n \in \{1, 2, \ldots, N\}$ denotes the number of a client).

Phase 2. Each client sends S_n^l and CPC_n^l learned at the l-th layer at Phase 1 to the server.

Phase 3. The server aggregates all local skeletons learned at the l-th layer. For a local skeleton S_n^l ($n \in \{1, 2, \ldots, N\}$) sent by a client at Phase 2, we use an edge score vector $\mathrm{EF}(n, k)$ ($k \in \{1, 2, \ldots, K\}$ denoting the number of a variable) to store the counts of an edge existing in S_n^l. Specifically, if S_n^l contains an edge between T and X_k ($X_k \in S_n^l$), $\mathrm{EF}(n, k) = 1$ holds, or $\mathrm{EF}(n, k) = 0$ holds. Then, the total score of the edge in the aggregated skeleton at the l-th layer is computed as follows:

$$\underset{k=1,2,\ldots,K}{\mathrm{EF}(:, k)} = \sum_{n=1}^{N} \mathrm{EF}(n, k). \tag{1}$$

Here, we use the edge score $\mathrm{EF}(:, k)$ to determine whether the edge between T and X_k exists in the aggregated skeleton at the l-th layer. Specifically, if an edge score $\mathrm{EF}(:, k)$ is bigger than a user-defined threshold ratio $\times N$, we keep this edge in the aggregated skeleton S_T^l; otherwise, we discard the edge, as outlined in (2), and finally obtain the aggregated skeleton of T (S_T^l) at the l-th layer.

$$\underset{k=1,2,\ldots,K}{\mathrm{EF}(:, k)} \begin{cases} \geqslant \text{ratio} \times N, & \text{keep the edge,} \\ < \text{ratio} \times N, & \text{delete the edge.} \end{cases} \tag{2}$$

Phase 4. When one of the following conditions is satisfied, the FLSke subroutine stops and obtains S_T^* and CPC_T^*. Otherwise, the server continues to send the aggregated skeleton S_T^l to each client as its initial skeleton at the $l+1$ layer and proceed with Phases 1–4 until the two conditions below are satisfied.

(i) The skeleton S_T^l aggregated from the skeletons $S_1^l, S_2^l, \ldots, S_n^l$ on the server at the l layer is the same as the aggregated skeleton S_T^{l-1} at the $l-1$ layer, i.e., the aggregated skeleton no longer changes;

(ii) The number of direct neighbors of T in the aggregated skeleton S_T^l equals or is less than l.

In summary, at each layer, each client sends their learned local skeleton of T to the server, and then the server aggregates all skeletons by calculating the number of the edges between T and the variable in all learned local skeletons (i.e., EF scores) without touching the original data of each client. If the EF score of an edge connected to T is bigger than a user-defined threshold, then the edge is kept in the aggregation skeleton. After the aggregation process, the server sends the aggregation skeleton to each client as a new initial skeleton for the next layer of learning during Phases 3 and 4. This federated skeleton learning and sharing strategy repeats until the conditions in Phase 4 are satisfied.

5.2 FLSori

The FLSori subroutine aims to orient edges in the local skeleton learned at the FLSke subroutine. When using conditional independence tests to orient edges, two steps are always adopted: identifying v-structures and using the Meek-rule for edge orientation, while identifying v-structures is the key step. However, for identifying v-structures of the local skeleton in the federated setting, we face the following two problems.

Firstly, it is challenging to obtain accurate information from the learned local skeleton for identifying v-structures in a federated setting. As shown in Figure 4(a), the local skeleton of T (i.e., S_T^{*1}) are

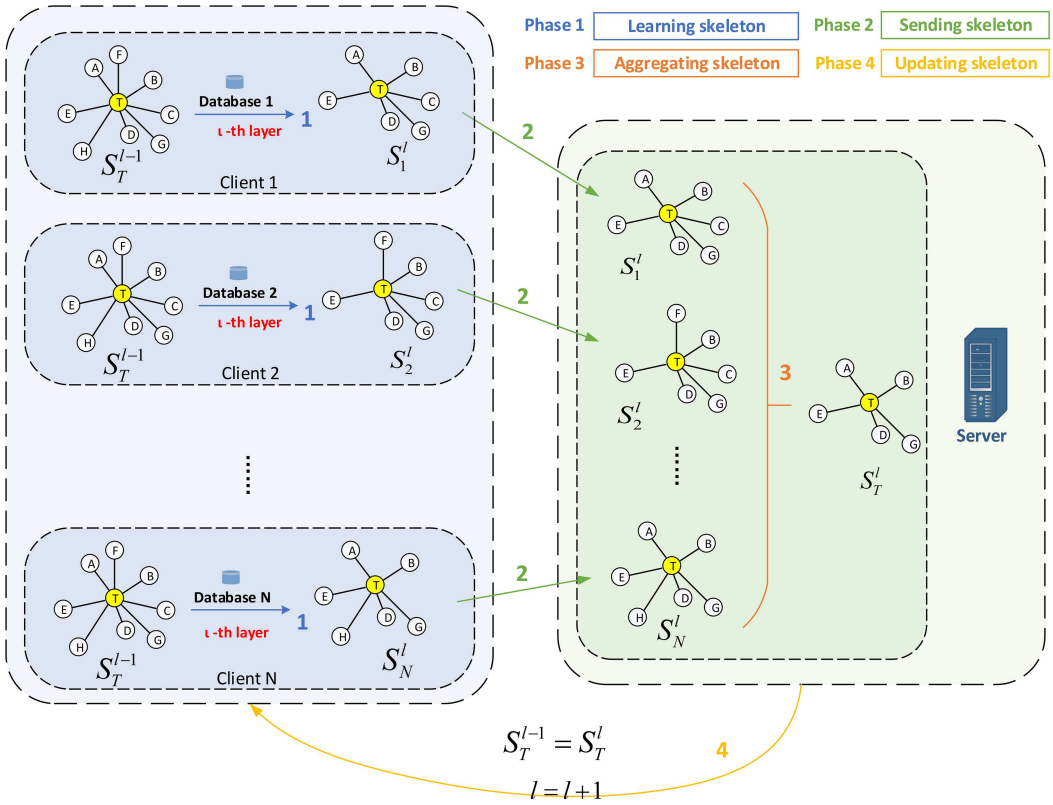


Figure 3 (Color online) Overview of the FLSke subroutines.

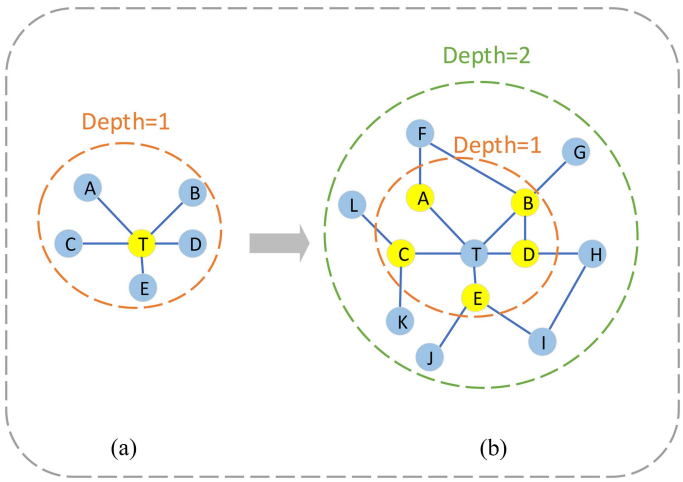


Figure 4 (Color online) (a) The local skeleton S_T^{*1} of T with a depth = 1 contains T and the variables in PC_T; (b) the local skeleton S_T^{*2} with a depth = 2 contains T, the variables in PC_T, and PC variables of the variables in PC_T.

formed only by its direct neighbors. When we identify whether an undirected triple X_i-T-X_j in the learned skeleton is a v-structure, we need to know the separation set of X_i and X_j, sepset(X_i, X_j). sepset(X_i, X_j) exists in $\mathrm{PC}_{X_i} \cup \mathrm{PC}_{X_j}$, but the learned skeleton of S_T^{*1} does not contain $\mathrm{PC}_{X_i} \cup \mathrm{PC}_{X_j}$ completely. In addition, for S_T^{*1} in Figure 4(a), we can observe that there is a triple: B-T-D, however, when we learn the depth-2 skeleton of T in Figure 4(b), we find that the triple B-T-D does not meet the conditions for v-structure determination because there is a direct edge between B and D.

Secondly, the sepset(X_i, X_j) set is not easy to compute even if we have the $\mathrm{PC}_{X_i} \cup \mathrm{PC}_{X_j}$ set. Since the separation set is computed in each client independently and each client has different data quality and data volumes, X_i and X_j may have different separation sets in different clients, thus it is difficult to achieve a consistent separation set of X_i and X_j among all clients.

To address the above problems, we propose two solutions: (1) federated extension of local causal skeleton for learning $\mathrm{PC}_{X_i} \cup \mathrm{PC}_{X_j}$, and (2) federated learning of consistent separation sets for identifying v-structures.

5.2.1 *Federated extension of local causal skeleton*

The phase extends the learned skeleton of T by learning the PC set of each variable of CPC_T^* in the skeleton S_T^{*1}. Firstly, for each variable $X_i \in \mathrm{CPC}_T^*$ in the learned local skeleton S_T^{*1}, we take it as the target variable and employ the FLSke subroutine described above to learn the local skeleton of X_i, $S_{X_i}^{*1}$, and then update the learned skeleton of T (called the depth-1 skeleton, S_T^{*1}, as shown in Figure 4(a)) to get the depth-2 skeleton S_T^{*2} of T, as shown in Figure 4(b). Then the depth-2 skeleton S_T^{*2} of T contains the PC set of each variable in PC_T, based on this information, we can find the separation sets at each client for each undirected triple X_i-T-X_j in the skeleton S_T^{*2}.

5.2.2 *Federated learning of consistent separation sets*

Although the separation sets are the by-products of the federated extension of the local causal skeleton phase, just as we discussed above, at this phase, for an undirected triple X_i-T-X_j, each client learns the PC set of X_i and X_j independently, and we may get different separation sets in different clients for X_i and X_j. Thus for an undirected triple X_i-T-X_j, how can we achieve a consistent separation set of X_i and X_j among all clients for accurate v-structure identification? To tackle this problem, we present a strategy for learning consistent separation sets which consists of four phases as shown in Figure 5.

Phase 1. For an undirected triple X_i-T-X_j in the learned depth-2 skeleton S_T^{*2} of T, the server sends the variable set $\mathrm{PC}_{X_i} \cup \mathrm{PC}_{X_j}$ instead of the entire learned skeleton to each client for learning the separation set of X_i and X_j.

Phase 2. Starting with the conditioning set size $l = 0$, each client independently finds all conditioning sets Z from $\mathrm{PC}_{X_i} \cup \mathrm{PC}_{X_j}$ where $|Z| = l$, ensuring $X_i \perp\!\!\!\perp X_j | Z$ (i.e., conditioning sets with p-value greater than α) and sends them to the sever. The server then picks the conditioning set with the highest p-value as the candidate consistent separation set. Subsequently, l is increased by 1, and each client finds all $|Z| = 1$ conditioning sets and sends them to the server, which again selects the highest p-value set as the candidate. This iterative process is repeated, increasing l each round, until l exceeds a predefined threshold or no conditioning sets are found at each client. Finally, the server selects the conditioning set with the highest p-value from all candidate consistent separation sets as the consistent separation set of X_i and X_j.

Assuming that the null hypothesis (H_0) of the CI test assumes independence between X_i and X_j within the skeleton S_T^*, the rationale behind Phase 2 is that according to the result in previous studies [37], the conditioning set with the highest p-value has the most possibility to make X_i and X_j independent and thus we can minimize the risk of finding wrong conditional independence by selecting the conditioning set with the maximum p-value as a consistent separation set [38, 39].

Phase 3. The server identifies the consistent separation set with the highest p-value from all candidate consistent separation sets, then uses the consistent separation set to determine whether the undirected triple X_i-T-X_j is a v-structure.

Phase 4. Based on the learned v-structures, we use the Meek-rule [26] to orient the remaining undirected edges in the depth-1 skeleton of T, S_T^{*1}, as many as possible on the server with two main rules: (1) no new v-structures being generated in S_T^{*1}, and (2) keeping acyclicity in S_T^{*1}.

5.3 FLEori

After orienting edges in the FLSori subroutine, if some edges in the depth-1 skeleton S_T^{*1} of T are still not oriented, how do we orient these edges in the federated setting? In this subsection, we propose the subroutine of FLEori to tackle this problem. In the FLEori subroutine, we examine all edges connected to T in its depth-1 skeleton, then save the undirected neighbors (if they exist) into a variable set Q. For instance, in Figure 6(a) the edge T-C is undirected, so $Q = \{C\}$. Then for each variable $X_i \in Q$, we implement the following steps to orient the edge between X_i and T.

Step 1. Taking $X_i \in Q$ as a target variable, based on the depth-1 skeleton $S_{X_i}^{*1}$, we employ the FLSke subroutine to get the PC set of the variables in the PC_{X_i}, obtaining the depth-2 skeleton $S_{X_i}^{*2}$. In Figure 6(a), the yellow nodes are the PC of C and we do not learn their PC sets yet. These yellow nodes

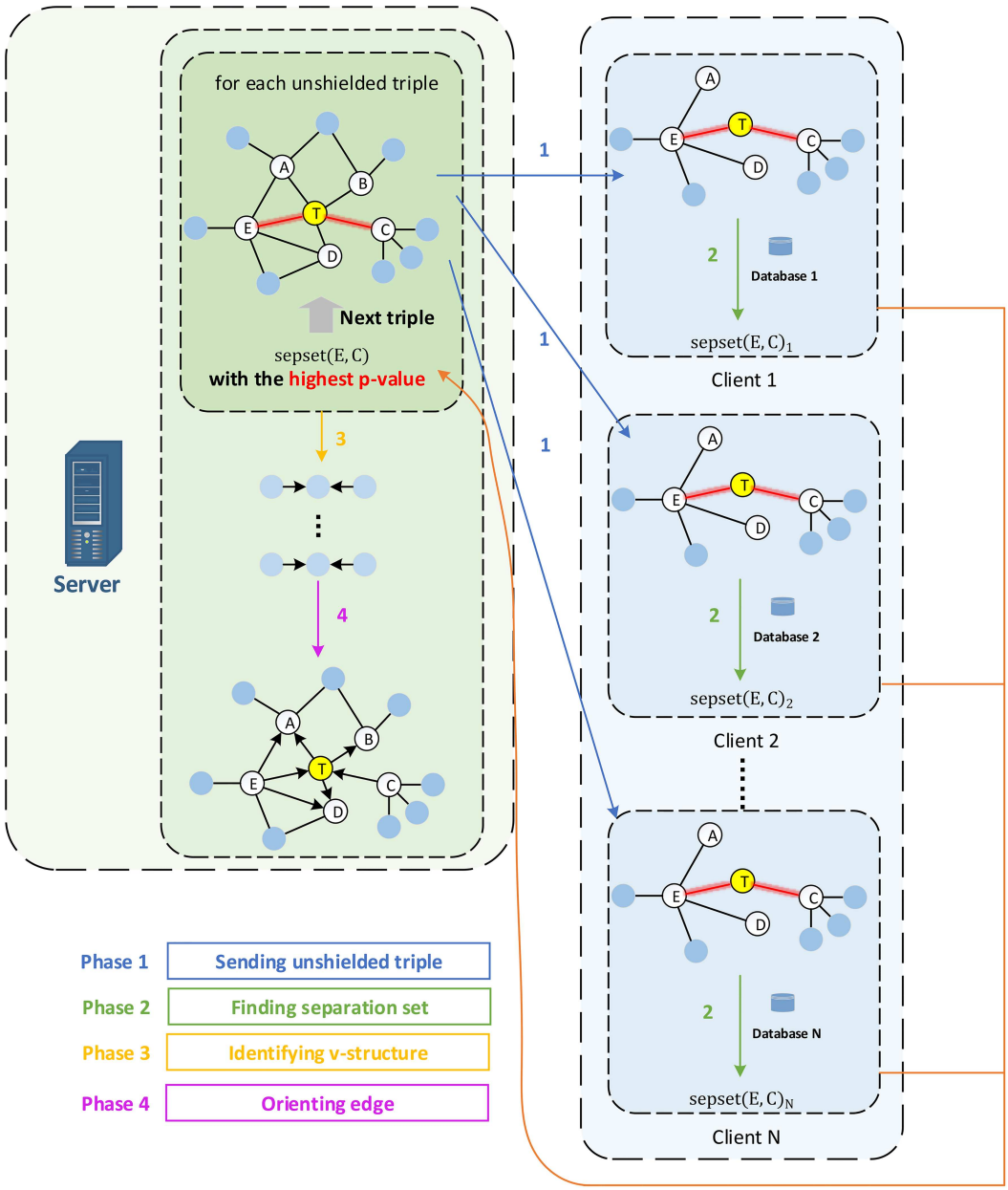


Figure 5 (Color online) Overview of federated learning of consistent separation sets.

and T, A, B form the depth-1 skeleton S_C^{*1}. At this step, we learn the PC set of the yellow nodes by the FLSke subroutine and obtain the depth-2 skeleton S_C^{*2} as shown in Figure 6(b).

Step 2. Using the obtained depth-2 skeleton $S_{X_i}^{*2}$ of X_i, at step 2 we update the depth-2 skeleton of T learned at the FLSori subroutine. Then the depth-2 skeleton $S_{X_i}^{*2}$ contains the PC set of each variable in PC_{X_i}, as same as the federated learning of consistent separation sets described above, we can compute the separation sets at each client for each undirected triple X_k-X_i-X_j in the depth-1 skeleton $S_{X_i}^{*1}$. For example, we use the depth-2 skeleton S_C^{*2} to update the skeleton of T, and obtain the updated skeleton of T as shown in Figure 6(b). Next, we compute the separation set on each client for undirected triples X_k-C-X_j in the depth-1 skeleton S_C^{*1} for identifying the consistent separation set.

Step 3. Based on the identified separation set of X_j and X_k for the undirected triple X_k-X_i-X_j, we determine whether X_k-X_i-X_j is a v-structure. If there are v-structures in the depth-2 skeleton $S_{X_i}^{*2}$, we use these v-structures to determine the edge direction of X_i and T. For example, in Figure 6(c), there exist three v-structures in the depth-2 skeleton S_C^{*2}, we use the v-structures and the Meek-rule to backtrack the edge direction of C and T, as shown in Figure 6(d).

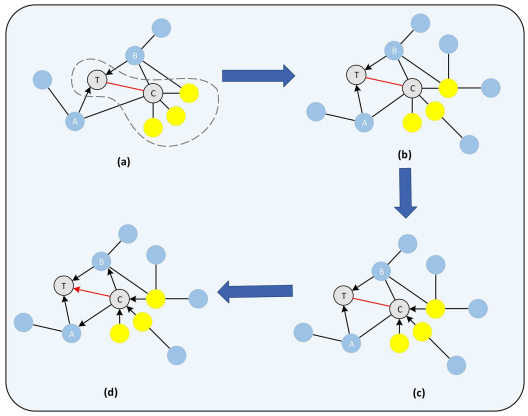


Figure 6 (Color online) Examples of the FLEori subroutines. (a) shows that the edge between T and C cannot be oriented in S_T^{*2}; (b) shows the skeleton extension to C and gets S_C^{*2}; (c) demonstrates the v-structures identified around C in S_C^{*2}; (d) shows that we backtrack the orientation of unoriented edges around T by the direction in S_C^{*1}.

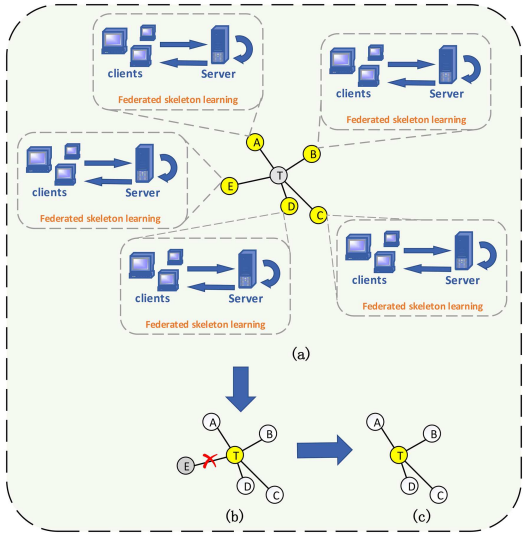


Figure 7 (Color online) Federated skeleton correction using the AND rule. (a) shows that the FLSke subroutine is performed on each PC variable of T; (b) shows the $T \notin \mathrm{PC}_E$ and this does not comply with the AND rule; (c) deletes E to correct the skeleton.

Step 4. To determine the edge direction of X_i and T, we use the Meek-rule to further orient the remaining undirected edges in the depth-1 skeleton of T; otherwise, step 4 continues for learning the depth-2 skeleton of yellow nodes (continues to expand outward one layer) to identify new v-structures until the edge of X_i and T is oriented or the expansion layers meet the threshold.

After each round of the above four steps, the FLEori subroutine examines whether undirected edges still exist in the depth-1 skeleton of T, S_T^{*1}, given that the edge between some variables in Q and T may be oriented after we orient the edge of X_i and T. If all edges in the depth-1 skeleton of T are oriented, FLEori stops; otherwise, steps 1 to 4 are repeated until one of the following conditions is satisfied:

(i) All edges around T in the local skeleton are oriented (in this case, FedLCS returns a DAG);

(ii) The depth of skeleton extension at step 4 exceeds a given threshold (the impact of the threshold on the performance of FedLCS has been analyzed in Appendix C).

If there are still undirected edges connected to T after the depth skeleton extension and orientation (i.e., FedLCS returns a CPDAG), we usually employ an acyclic constraint technique, as described by Spirtes et al. [40], to randomly orient the undirected edges in S_T^{*1}.

To enhance the correctness of local structure learning, the FedLCS algorithm uses the AND rule to further correct the learned local skeleton in the three subroutines. According to the AND rule, if $X_i \in \mathrm{PC}_T$ and $T \in \mathrm{PC}_{X_i}$, then there is an edge between X_i and T. In Figure 7, we first take T as the target variable and learn that the $\mathrm{PC}_T = \{A, B, C, D, E\}$; next, we verify that for each $X_i \in \mathrm{PC}_T$, whether $T \in \mathrm{PC}_{X_i}$. Second, FedLCS takes each variable in $\{A, B, C, D, E\}$ as the target variable successively and employs the FLSke subroutine to learn their PC set, as shown in "federated skeleton learning" part in Figure 7(a), assuming that the learning result is that $T \notin \mathrm{PC}_E$ holds while T is in other variables' PC set. Therefore, E is deleted from S_T^{*1} (i.e., PC_T). At this point, the skeleton of T, S_T^{*1} is determined.

6 Experiments

This section presents a comprehensive experiment to demonstrate the effectiveness of the proposed FedLCS algorithm. The organization of this section is as follows. Subsection 6.1 provides the experimental settings. Subsection 6.2 shows the performance of FedLCS against its rivals across three types of datasets. Due to space limitations, we put the results on the Sachs dataset into Appendix D. Subsection 6.3 compares the performance of FedLCS with FedPC and NOTEARS-ADMM, highlighting the advantages of the proposed FedLCS algorithm. Additionally, we provide a verification experiment for

the consistent separation set strategy of FedLCS in Appendix B, and we also give the experiments for determining the necessary parameters in Appendix C.

6.1 Experimental settings

6.1.1 *Evaluation metrics*

We evaluated the performance of FedLCS and its nine rival algorithms from two aspects: structure correctness and structure error, which are measured by the $F1$ and SHD (structural hamming distance) metrics, respectively.

• Structure correctness. $F1 = \frac{(2\times\text{Recall}\times\text{Precision})}{(\text{Recall}+\text{Precision})}$, the "Precision" metric denotes the number of correctly predicted arrowheads in the output divided by the number of edges in the output of an algorithm, while the "Recall" metric represents the number of correctly predicted arrowheads in the output divided by the number of true arrowheads in a test DAG. The larger value of $F1$ is better.

• Structure error. The SHD value is the number of total error edges, containing undirected, reverse, missing, and extra edges. The smaller value of SHD is better.

We run FedLCS for learning the local causal structure of all variables and we obtain $F1^1, F1^2, \ldots, F1^K$ (where K represents the number of variables). The average $F1$ of all variables, $\overline{F1} = \{F1^1 + F1^2 + \cdots + F1^K\}/K$ is used as the final value of $F1$ for evaluating the structure correctness of FedLCS. As for the SHD metric, we do the same process for it.

6.1.2 *Comparison methods*

Considering that no existing study exists for LCS in a federated setting, to validate our proposed FedLCS, we design the nine FedLCS algorithms based on the well-established PCD-by-PCD, MB-by-MB, CMB, and FS-LCS algorithms, PCD-by-PCD-best, PCD-by-PCD-avg, MB-by-MB-best, MB-by-MB-avg, CMB-best, CMB-avg, FS-LCS-best, FS-LCS-avg. All methods under comparison are implemented under the same datasets and the same number of clients.

• CMB-Avg. We first run the CMB algorithm on each client for obtaining local causal structures of a target variable independently without any information sharing and then compute the averaging $F1$ and SHD values ($\overline{F1}$ and $\overline{\text{SHD}}$) of all the computed local causal structures as the final output of MB-Avg. The main idea of PCD-by PCD-Avg, MB-by-MB-Avg, and FS-LCS-Avg is the same as CMB-Avg, and we do not describe them at all.

• CMB-Best. We first run the CMB algorithm on each client for obtaining local causal structures of a target variable independently without any information sharing, then obtain $\{F1_{D_1}, F1_{D_2}, \ldots, F1_{D_N}\}$ and $\{\text{SHD}_{D_1}, \text{SHD}_{D_2}, \ldots, \text{SHD}_{D_N}\}$ from N isolated clients respectively, and we select the largest $F1$ value and the smallest SHD value as the final results of CMB-Best. The main idea of PCD-by PCD-Best, MB-by-MB-Best, and FS-LCS-Best is the same as CMB-Best.

• LCS-All. We aggregate all datasets to a single client, that is, under a non-federated environment, we perform LCS based on all clients' datasets.

Additionally, we compare FedLCS with the advanced federated global causal structure algorithms, FedPC and NOTEARS-ADMM. Since FedPC and NOTEARS-ADMM obtain a global causal structure, the evaluation metrics cannot be directly compared with those of the local causal structure, so we extract the local causal structure of each variable from the learned global causal structure and compare it with the true DAG to obtain the $F1$ and the SHD metrics. Finally, by comparing $\overline{F1}_{\text{FedPC}}$, $\overline{F1}_{\text{NOTEARS-ADMM}}$, $\overline{F1}_{\text{FedLCS}}$, $\overline{\text{SHD}}_{\text{FedPC}}$, $\overline{\text{SHD}}_{\text{NOTEARS-ADMM}}$, and $\overline{\text{SHD}}_{\text{FedLCS}}$, we can understand the performance differences between the two algorithms.

6.1.3 *Datasets*

For the proposed FedLCS algorithm, we compare it with the method described in Subsection 6.1.2 on six BN datasets, six synthetic datasets, and a real dataset provided by [41]. Both BN datasets and synthetic datasets consist of 5000 samples.

In the federated setting, the data samples are randomly distributed across the clients, and each client has the same set of variables but a unique and non-repeating set of samples. We assume there are N clients and N lies in $\{3, 5, 10, 15\}$. And we set each client to contain at least $5000/\{N \times 2\}$ data samples. The detailed information on BNs is shown in Table 3.

Table 3 Summary of benchmark BNs. Details of these benchmark BNs are publicly available at http://www.bnlearn.com/bnrepository/.

Network	Number of vars	Number of edges	Max in/out-degree	Min/max |PCset|	Average degree
Insurance	27	52	3/7	1/9	3.85
Alarm	37	46	4/5	1/6	2.49
Win95pts	76	112	7/10	1/10	2.95
Andes	223	338	6/12	0/12	3.03
Pigs	441	592	2/39	1/41	2.68
Gene	801	972	4/10	0/11	2.42

Table 4 Random sampling of the Gene dataset.

Size of |PCset|	Number of nodes	Size of samples
0	45	0
1–3	591	28
4–7	155	10
8–11	10	2

For the high-dimensional Gene dataset, we extracted some variables for local structure learning. We randomly selected a total of 40 variables from each interval, as shown in Table 4.

For the linear synthetic datasets, we generated six continuous datasets using an open-source toolkit [42] with the number of variables 10, 20, 50, 100, 300, and 400, respectively. The generative process employs a linear causal mechanism represented as follows:

$$y = XW + \times E, \tag{3}$$

where $+\times$ denotes either addition or multiplication, X denotes the vector of causes, and E represents the noise variable accounting for all unobserved variables.

The continuous real dataset, Sachs [43], with 853 samples of a protein signaling network, expresses levels of different proteins and phospholipids in human cells and is considered a benchmark graphical model with 11 nodes (cell types) and 17 edges.

6.1.4 *Implementation details*

All experiments were conducted on a computer with Inter Core i7-8700 3.70 GHz CPU and 48 GB of memory. All methods were implemented in MATLAB. The significance level for the CI test was set to 0.05. According to the parameter experiments, the maximum layer for FedLCS was set to 6, the ratio threshold for client voting was set to 0.4, and the value of the extension depth for the federated extension-and-backtracking orientation was set to 2 (i.e., the default value).

6.2 Results on benchmark and synthetic datasets

This subsection presents the F1, and SHD of FedLCS and its rivals, respectively.

Structure correctness. The first rows of sub-figures in Figures 8 and 9 depict the $F1$ of FedLCS and its rivals using six benchmark BN datasets and six synthetic datasets. Evidently, in most cases, FedLCS outperforms that of its rivals in terms of $F1$. The results are explained as follows.

The rivals show that in scenarios with insufficient and private data, non-federated local learning algorithms can only learn on isolated clients. Their processes are independent and confidential, with only the final results being publicized for optimization through averaging or choosing the best results. However, the FedLCS algorithm can exchange local skeletons and separation sets information during the learning process, significantly compensating for the disadvantages of the issues of data insufficiency and different data quality across different clients. During the skeleton learning phase, the aggregation strategy largely corrects erroneous edges. Moreover, in the orientation process, a consistent separation set closer to the true separation set guarantees identifying accurate v-structures.

As the number of clients increases, the volume of data samples on each client decreases. LCS-All aggregates all datasets onto a single platform for local structure learning. Evidently, under such ideal conditions of abundant data, the $F1$ metric on the BN datasets generally reaches the optimum. However, the $F1$ of LCS-All on the synthetic continuous datasets is only average and is not as good as that of

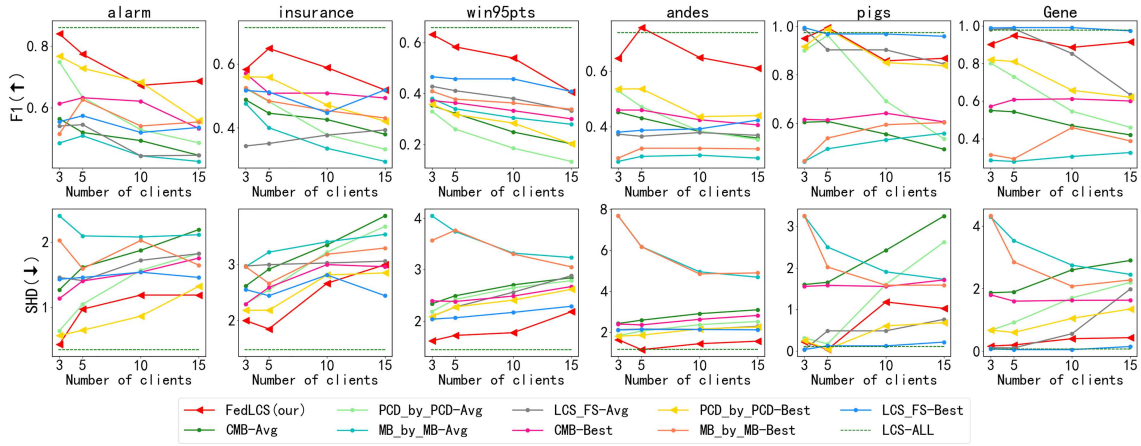


Figure 8 (Color online) Results on benchmark BN datasets.

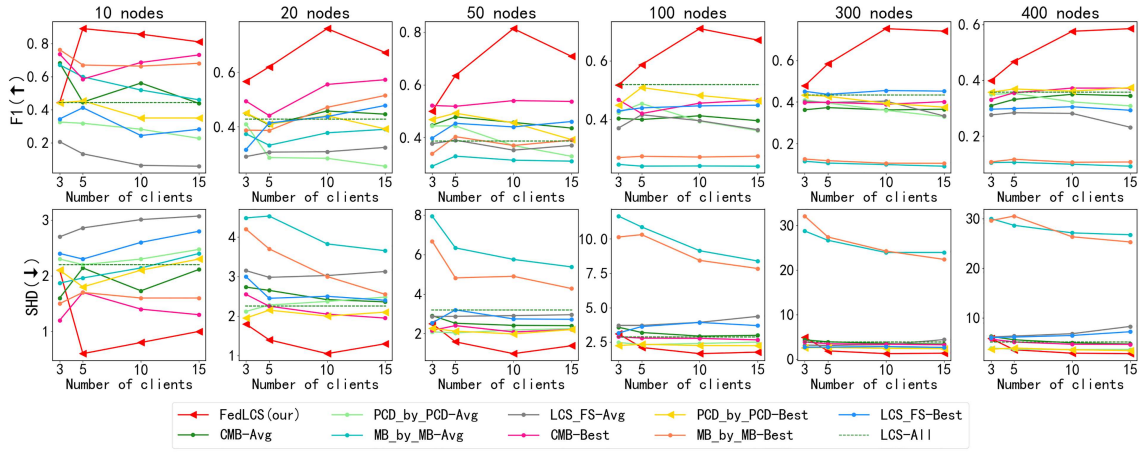


Figure 9 (Color online) Results on synthetic datasets.

FedLCS. Considering the results of all rivals on the synthetic datasets, the overall trend does not change significantly with the change in the number of clients (as compared to the trend of results on the BN datasets).

In a non-federated environment, the performance of LCS-All may be insensitive when the synthetic data sample size is beyond a certain threshold (i.e., increasing data samples does not improve the performance of LCA-ALL). This phenomenon can be attributed to the fact that synthetic datasets are often generated based on linear causal mechanisms. Accordingly, the clients' collaborations may promote the performance of the CSL algorithms in a federated setting.

Structure error. The second rows of sub-figures in Figures 8 and 9 display the SHD of FedLCS and its rivals using six benchmark datasets and six synthetic datasets. FedLCS achieves the lowest SHD on almost all datasets, validating the superiority of FedLCS.

The values of SHD of LCS-All on discrete datasets are superior to FedLCS, while the situation is not the same on the synthetic datasets. The possible explanation is just discussed above that the aggregation strategy in the federated setting may help FedLCS and its eight rivals improve their performance.

It can be observed that on benchmark datasets, the SHD increases with the increasing number of clients. This is because as the number of clients increases, the number of data samples on each client decreases, leading to insufficient data samples on each client. This data insufficiency leads to unreliable CI tests, resulting in incorrect local skeletons and inaccurate separation sets.

On continuous synthetic datasets, the SHD value does not show significant changes as the number of clients changes. This is because the quality of synthetic datasets is relatively stable, thus smaller amounts of data can still allow local structure learning algorithms to converge quickly.

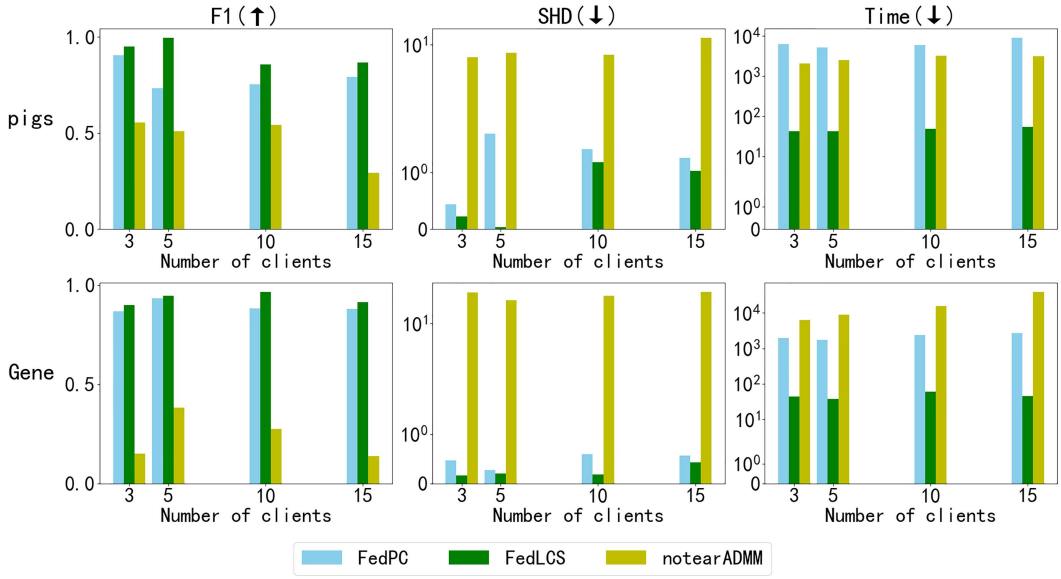


Figure 10 (Color online) Comparison results of FedLCS, FedPC, and NOTEARS-ADMM.

6.3 Comparison of FedPC, NOTEARS-ADMM and FedLCS

When focusing solely on the local causal structure around a specific variable in a large-scale causal structure, global structure learning algorithms inevitably increase unnecessary time costs. To illustrate this, we compare FedLCS with two global structure learning algorithms, NOTEARS-ADMM and FedPC, using two high-dimensional datasets: Pigs and Gene. For FedPC and NOTEARS-ADMM, we first learn the global causal structure and extract the local causal structures for each variable from the global model. These extracted local structures are compared with the true DAG to obtain performance metrics. FedLCS learns the local causal structure for each variable independently and compares it directly with the true DAG to obtain the metrics.

The results shown in Figure 10 indicate that FedLCS outperforms FedPC and NOTEARS-ADMM on both the structural correctness and the structural error. For running time, FedLCS significantly outperforms FedPC and NOTEARS-ADMM. This superiority demonstrates that on high-dimensional datasets, the local learning algorithm FedLCS is better than the global learning methods FedPC and NOTEARS-ADMM.

7 Conclusion and future work

In this study, we address the challenge of learning local causal structures within a federated setting, while emphasizing data privacy protection. We introduce a novel algorithm, FedLCS, specifically designed for this purpose. FedLCS comprises three key subroutines: FLSke, FLSori, and FLEori. Initially, FLSke derives the local causal skeleton of a target variable, FLSori then identifies v-structures within this skeleton to conduct edge orientation, and FLEori orients the remaining undirected edges. To the best of our knowledge, FedLCS represents the first effort to solve the problem of LCS in a federated environment.

We analyze the limitations of this study and suggest possible future directions. In the skeleton aggregation phase, we assigned equal decision-making weight to each client for edge voting. However, in real scenarios, data quality can vary among clients, which may make this approach inadequate for accurately reflecting each client's actual influence. Future research could explore methods to evaluate and score the skeleton results from each client at each layer, allowing for differential weighting based on data quality. Furthermore, our current approach uses an edge score vector strategy, which involves calculating for edge selection. Future work could focus on systematically exploring different structure aggregation strategies to enhance the accuracy and robustness of the LCS process.

Acknowledgements This work was supported by National Science and Technology Major Project of China (Grant No. 2020AAA-0106100) and National Natural Science Foundation of China (Grant Nos. 62376087, 62306002).

Supporting information Appendixes A–E. The supporting information is available online at info.scichina.com and link.springer.com. The supporting materials are published as submitted, without typesetting or editing. The responsibility for scientific accuracy and content remains entirely with the authors.

References

1 Cai R C, Qiao J, Zhang Z J, et al. Self: structural equational likelihood framework for causal discovery. In: Proceedings of the 38th AAAI Conference on Artificial Intelligence, Vancouver, 2018. 1787–1794
2 Kuang K, Li L, Geng Z, et al. Causal inference. Engineering, 2020, 6: 253–263
3 Pearl J. Causality. Cambridge: Cambridge University Press, 2009
4 Liang J, Wang J, Yu G, et al. Directed acyclic graph learning on attributed heterogeneous network. IEEE Trans Knowl Data Eng, 2023, 35: 10845–10856
5 Cai R, Wu S, Qiao J, et al. THPs: topological Hawkes processes for learning causal structure on event sequences. IEEE Trans Neural Netw Learn Syst, 2022, 35: 479–493
6 Guo X, Yu K, Liu L, et al. Adaptive skeleton construction for accurate DAG learning. IEEE Trans Knowl Data Eng, 2023, 35: 10526–10539
7 Zheng X, Aragam B, Ravikumar P K, et al. Dags with no tears: continuous optimization for structure learning. In: Proceedings of the 32nd International Conference on Neural Information Processing Systems, Montreal, 2018. 9492–9503
8 Yang S, Wang H, Yu K, et al. Towards efficient local causal structure learning. IEEE Trans Big Data, 2022, 8: 1592–1609
9 Gao T, Ji Q. Local causal discovery of direct causes and effects. In: Proceedings of the 28th International Conference on Neural Information Processing Systems, Montreal, 2015. 2512–2520
10 Scanagatta M, Corani G, de Campos C P, et al. Learning treewidth-bounded Bayesian networks with thousands of variables. In: Proceedings of 30th Annual Conference on Neural Information Processing Systems, Barcelona, 2016. 1470–1478
11 Yu K, Liu L, Li J. A unified view of causal and non-causal feature selection. ACM Trans Knowl Discov Data, 2021, 15: 1–46
12 Cui P, Athey S. Stable learning establishes some common ground between causal inference and machine learning. Nat Mach Intell, 2022, 4: 110–115
13 Lu Y, Huang X, Dai Y, et al. Blockchain and federated learning for privacy-preserved data sharing in industrial IoT. IEEE Trans Ind Inf, 2019, 16: 4177–4186
14 Li Q, Wen Z, Wu Z, et al. A survey on federated learning systems: vision, hype and reality for data privacy and protection. IEEE Trans Knowl Data Eng, 2023, 35: 3347–3366
15 Arasteh S T, Ziller A, Kuhl C, et al. Private, fair and accurate: training large-scale, privacy-preserving AI models in medical imaging. 2023. ArXiv:2302.01622
16 Di Minin E, Fink C, Hausmann A, et al. How to address data privacy concerns when using social media data in conservation science. Conserv Biol, 2021, 35: 437–446
17 Zhang J, Chen B, Zhao Y, et al. Data security and privacy-preserving in edge computing paradigm: survey and open issues. IEEE Access, 2018, 6: 18209–18237
18 Li L, Fan Y, Tse M, et al. A review of applications in federated learning. Comput Indust Eng, 2020, 149: 106854
19 Zhang C, Xie Y, Bai H, et al. A survey on federated learning. Knowl-Based Syst, 2021, 216: 106775
20 Ng I, Zhang K. Towards federated Bayesian network structure learning with continuous optimization. In: Proceedings of the 25th International Conference on Artificial Intelligence and Statistics, 2022. 8095–8111
21 Gao E D, Chen J J, Shen L, et al. FedDAG: federated DAG structure learning. 2022. ArXiv:2112.03555v3
22 Huang J L, Guo X J, Yu K, et al. Towards privacy-aware causal structure learning in federated setting. 2023. ArXiv:2211.06919
23 Tang X Y, Guo S, Zhang J, et al. Learning personalized causally invariant representations for heterogeneous federated clients. In: Proceedings of the 12th International Conference on Learning Representations, Vienna, 2024
24 Yu K, Guo X, Liu L, et al. Causality-based feature selection. ACM Comput Surv, 2020, 53: 1–36
25 Tsamardinos I, Aliferis C F, Statnikov A. Time and sample efficient discovery of Markov blankets and direct causal relations. In: Proceedings of the 9th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, 2003. 673–678
26 Meek C. Causal inference and causal explanation with background knowledge. In: Proceedings of the 11th Conference on Uncertainty in Artificial Intelligence, Montreal, 1995. 403–410
27 Tsamardinos I, Aliferis C F, Statnikov A R, et al. Algorithms for large scale Markov blanket discovery. In: Proceedings of the 16th International Florida Artificial Intelligence Research Society Conference, Florida, 2003. 376–381
28 Guo X, Yu K, Cao F, et al. Error-aware Markov blanket learning for causal feature selection. Inf Sci, 2022, 589: 849–877
29 Wang C, Zhou Y, Zhao Q, et al. Discovering and orienting the edges connected to a target variable in a DAG via a sequential local learning approach. Comput Stat Data Anal, 2014, 77: 252–266
30 Ling Z, Yu K, Wang H, et al. Using feature selection for local causal structure learning. IEEE Trans Emerg Top Comput Intell, 2020, 5: 530–540
31 Ling Z, Yu K, Liu L, et al. PSL: an algorithm for partial Bayesian network structure learning. ACM Trans Knowl Discov Data, 2022, 16: 1–25
32 McMahan B, Moore E, Ramage D, et al. Communication-efficient learning of deep networks from decentralized data. In: Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, Fort Lauderdale, 2017. 1273–1282
33 Kairouz P, McMahan H B, Avent B, et al. Advances and open problems in federated learning. FNT Mach Learn, 2021, 14: 1–210
34 Li T, Sahu A K, Talwalkar A, et al. Federated learning: challenges, methods, and future directions. IEEE Signal Process Mag, 2020, 37: 50–60
35 Pearl J. Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference. San Francisco: Morgan Kaufmann Publishers Inc., 1988
36 Li J Y, Liu L, Le T D, et al. Local causal discovery with a simple PC algorithm. In: Practical Approaches to Causal Relationship Exploration. Berlin: Springer, 2015. 9–21
37 Ramsey J, Improving accuracy and scalability of the PC algorithm by maximizing P-value. 2016. ArXiv:1610.00378
38 Li H H, Cabeli V, Sella N, et al. Constraint-based causal structure learning with consistent separating sets. In: Proceedings of the 32nd Annual Conference on Neural Information Processing Systems, Vancouver, 2019. 14257–14266
39 Colombo D, Maathuis M H. Order-independent constraint-based causal structure learning. J Mach Learn Res, 2014, 15: 3741–3782
40 Spirtes P, Glymour C N, Scheines R. Causation, Prediction, and Search. Cambridge: MIT Press, 2000
41 Tsamardinos I, Brown L E, Aliferis C F. The max-min hill-climbing Bayesian network structure learning algorithm. Mach Learn, 2006, 65: 31–78
42 Kalainathan D, Goudet O, Dutta R. Causal discovery toolbox: uncovering causal relationships in Python. J Mach Learn Res, 2020, 21: 1406–1410
43 Sachs K, Perez O, Pe'er D, et al. Causal protein-signaling networks derived from multiparameter single-cell data. Science, 2005, 308: 523–529