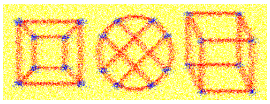
Journal of Graph Algorithms and Applications
http://jgaa.info/ vol. 30, no. 1, pp. 253–272 (2026)
DOI: 10.7155/jgaa.v30i1.3236

Polynomial-Time Derivation of Optimal k-Tree Topology from Markov Networks

Fereshteh R. Dastjerdi[1] *Liming Cai*[1]

[1]School of Computing, University of Georgia, Athens, GA 30602

Submitted: April 2024 Accepted: May 2026 Published: June 2026

Article type: Regular paper Communicated by: E. Di Giacomo and M. Nöllenburg

Abstract. Characterization of joint probability distribution for large networks of random variables remains a challenging task in data science. Probabilistic graph approximation with simple topologies has practically been resorted to; typically the tree topology makes joint probability computation much simpler and can be effective for statistical inference on insufficient data. However, to characterize network components where multiple variables cooperate closely to influence others, model topologies beyond a tree are needed, which unfortunately are infeasible to acquire. In particular, our previous work has related the approximation of Markov networks of tree-width k with the minimum information loss (termed optimal approximation) closely to the graph-theoretic problem of finding maximum spanning k-tree (MSkT), which is a provably intractable task.

This paper investigates optimal approximation of Markov networks with k-tree topology that retains some designated underlying subgraph. Such a subgraph may encode certain background information that arises in scientific applications, for example, about a known significant pathway in gene networks or the indispensable backbone connectivity in the residue interaction graphs for a biomolecule 3D structure. In particular, it is proved that the β-retaining MSkT problem, for several classes β of graphs (i.e., families of subgraphs that are required to be retained, such as Hamiltonian paths), admits $O(n^{k+1})$-time algorithms for fixed $k \geq 1$. These β-retaining MSkT algorithms offer efficient solutions for approximation of Markov networks with k-tree topology in situations where certain persistent structural information needs to be preserved.

1 Introduction

To accurately model complex real world systems that exhibit relationships among random variables, it is often crucial to compute the joint probability distribution of a set of random variables [2, 30]. The joint distribution function $P(\mathbf{X})$, for set $\mathbf{X}$ of random variables, serves as a fundamental

E-mail addresses: fereshtehrabiei@gmail.com (Fereshteh R. Dastjerdi) liming@uga.edu (Liming Cai)

basis for a wide range of statistical and probabilistic modeling techniques employed in many fields, such as communication, pattern recognition, machine learning, information retrieval, databases, and learning systems [16, 17, 36]. Distribution $P(\mathbf{X})$ captures the relationships and dependencies between different variables or events within the system. It provides insights into the likelihood of specific outcomes, helps identify patterns and correlations, and enables prediction of unseen data [33, 34]. It can be challenging, and sometimes even impossible within a reasonable time frame, to determine a single probability distribution due to the complex relationships between random variables. For example, to fully specify the n^{th}-order probability distribution $P(\mathbf{X})$ of an event space defined by n binary random variables, the knowledge of $2^n - 1$ probability values would be required. Indeed, the exponential size of the event space makes the exact learning of $P(\mathbf{X})$ to be an intractable task [11, 24, 33]. The challenge becomes even greater when only a limited number of data samples are at hand [12].

Various approaches to approximate $P(\mathbf{X})$ have been proposed for situations where the information of random variables is given with a limited number of samples [12, 21, 22, 29, 33–35, 47]. One common approach is to approximate the n^{th}-order $P(\mathbf{X})$ using a lower-order probability function. Lower-order approximations encompass the m^{th}-order approximations for $m \ll n$. These approximations often involve simplifying assumptions, such as statistical independence or other constraints imposed on dependency relationships between the random variables. By employing these techniques, it becomes possible to compute a specific joint probability distribution that aligns with the available data. Their effectiveness is contingent upon the accuracy of approximations in capturing the true distribution $P(\mathbf{X})$. In a study by King *et al.* [26], the authors specifically focused on a second-order approximation ($m = 2$) to approximate a multivariate probabilistic model. The second-order approximation considers only the pairwise dependency between variables, reducing the computational complexity of calculating $P(X)$ to $O(n^2)$.

The seminal work of Chow and Liu [12] has made a significant contribution to the field of approximating $P(\mathbf{X})$ using a second-order approximation method with a tree-topology known as the Chow-Liu tree. Technically, this approximation method constructs a maximum spanning tree of the network whose topology guarantees to minimize information loss in approximating $P(\mathbf{X})$, as measured by the Kullback-Leibler divergence [31]. The Chow-Liu method has been proven both practical and computationally efficient, even with limited data availability [4, 22]. In various applications, in the context of either physical or artificial systems, such a tree structure often serves as an effective model for approximating the underlying system [49]. As a result, the Chow-Liu tree is widely employed as a flexible and interpretable model across different tasks including dimensionality reduction [51], systems biology [**?**], gene differential analysis [47], discrete speech recognition [21], data classification and clustering tasks [7], and time-series data analyses [46].

However, an approximation of probability distribution $P(\mathbf{X})$ with tree topology may be below the desired or expected level of performance when dealing with distributions of high-dimensional data. In many real world scenarios, dependencies and relationships between variables are often more complex than what a tree structure can capture [22, 25, 50]. Specifically, the tree topology fails to deal with the scenario where one variable is an effect caused by collaborative work of two or more other variables [5]. Even in real world applications that exhibit tree-like (sparse) network structures [44], there exist critical components that may not be a tree structure. One such example is a bottleneck component of several genes working together to influence a couple of downstream genes in a cancerous pathway within a gene-network [20]. Networks containing such locally coupled components while demonstrating global sparsity are common, e.g., a social network containing small groups of highly related people, an article citation network for advancement of specific research topic, and a biological network including a cluster of interacting proteins [23, 37].

Hence, approximation of probability distribution $P(\mathbf{X})$ with sparse, non-tree graph topology is desirable; yet measuring the quality of approximation requires an accurate definition of network sparsity. Sparsity of a graph is often loosely defined by the ratio of the number of edges to the number of vertices in the graph. Typically a graph $G = (V, E)$ is sparse if $\frac{|E|}{|V|} = O(1)$, i.e., the number of edges is at most linearly related to the number of vertices. This metric, while intuitive, may not account differences in global topology among such graphs. For example, "tree-like" graphs bear very different global connectivity from those "grid-like" graphs, in spite of both having a linear number of edges. The global connectivity of graphs can have an immediate impact on the tractability of graph optimization problems, which can be well quantified with the notion of graph "tree width". Tree-width measures how tree-like a graph is and it is intimately related to the notion of k-tree [3, 39, 41]. Indeed, under various settings, investigations have been carried out on approximation of $P(\mathbf{X})$ with topology of graphs of small tree-width [25, 45, 48]. In particular, our previous work established a connection between the problem of approximating $P(\mathbf{X})$ with a Markov network of tree-width k and the graph-theoretic task of finding a maximum spanning k-tree (MSkT) in the underlying network graph [8]. Unfortunately, this task is computationally intractable for all fixed $k \geq 2$ [6].

In this paper, we focus on developing efficient algorithms for approximating Markov networks with tree-width bounded by k in a way that minimizes information loss—a task we refer to as *optimal approximation*. We study the problem under natural condition arising from applications, specifically when the desired topology of tree-width k retains certain essential subnet in the underlying Markov network. We coin this the problem β-retaining MSkT, which finds a maximum spanning k-tree that retains a subgraph (of class β) designated in the input graph. We prove that for class β of bounded degree spanning trees, the β-retaining MSkT problem can be solved in polynomial-time $O(n^{k+1})$ on graphs of n vertices for every fixed $k \geq 1$. To show that the achieved time complexity upper bound is likely optimal, we place our work in the realm of the parameterized complexity theory [14] that studies time complexity of algorithms and problems in terms of a special parameter k along with the input size n. In particular, we are able to reduce the classical graph problem k-Clique to β-retaining MSkT with a transformation that preserves the parameter value k exactly. Problem k-Clique is notoriously difficult to admit algorithms of time $O(n^{k-\epsilon})$, for $\epsilon > 0$ [1, 32] and provably $W[1]$-hard (i.e., unlikely to admit time-$O(t(k)n^c)$ for constant c and function t in k only [14]). Our presented transformation passes these difficulties of k-Clique onto problem β-retaining MSkT.

The algorithms for β-retaining MSkT solving optimal Markov network approximation are also practically useful for solving other scientific problems. In particular, since Hamiltonian paths constitutes a special class of bounded-degree spanning trees, the algorithms can be tailored to solving problem *Hamiltonian path-retaining* MSkT. This is especially useful for revealing hidden higher-order relationships over a set of random variables that possess an indispensable total order relation, e.g., time series data, linguistic sentences, and biomolecule sequences. Indeed, our work has inspired a general framework for efficient and accurate prediction of biomolecule 3D structures. According to this framework, on a given molecule consisting of an ordered sequence of residues, an underlying probabilistic graph can be formulated in which random variables represent residues, edges for possible interactions between residues, and "backbone edges" connecting neighboring residues on the sequence. The objective of the 3D structure prediction problem is to find a spanning 3-tree, which maximizes sum of mutual information of all involved 4-cliques in the 3-tree. Here the mutual information of four variables in a 4-clique is defined based on its probabilistic mapping to a tetrahedron motif involving 4 residues in the biomolecule, where tetrahedrons are geometric building blocks of the 3D structure [8].

2 Preliminaries

In this section, we introduce the notions of Markov networks, k-tree, tree-width, and Markov k-tree, which play essential roles in discussions throughout the paper.

2.1 Markov Networks

A finite system comprising n random variables is denoted as $\mathbf{X} = \{X_1, \ldots, X_n\}$, with the joint probability distribution function $P(\mathbf{X})$. While $P(\mathbf{X})$ may offer insights into dependence relationships among the variables, it is often not known due potentially high-dimensional relationships among the variables. The task we are interested in is to approximate the joint probability distribution $P(\mathbf{X})$ with $P_G(\mathbf{X})$, a joint distribution of $\mathbf{X}$ in their binary relationships characterized by graph G. We call G the *topology* of $P_G(\mathbf{X})$.

Definition 1. Let $G = (\mathbf{X}, E)$ be a undirected probabilistic graph with vertex set $\mathbf{X}$ consisting of random variables and edge set $E \subseteq \mathbf{X} \times \mathbf{X}$. Graph G is called a *Markov network* if, for the joint probability distribution $P_G(\mathbf{X})$,

$$(X_i, X_j) \notin E \Longrightarrow P_G(X_i, X_j|\mathbf{X}\backslash\{X_i, X_j\}) = P_G(X_i|\mathbf{X}\backslash\{X_i, X_j\})P_G(X_j|\mathbf{X}\backslash\{X_i, X_j\}). \quad (1)$$

That is, variables X_i and X_j without an edge between them are conditionally independent given the rest of the variables in the graph.

The property in (1) is called the *pairwise Markov property*. It is equivalent to the following presumably stronger condition, called *global Markov property* as long as the distribution $P_G(\mathbf{X})$ are always positive [18]. Specifically, for subsets of variables $\mathbf{Y}, \mathbf{Z}, \mathbf{S} \subseteq \mathbf{X}$ and separator $\mathbf{S}$ of $\mathbf{Y}$ and $\mathbf{Z}$ of graph G,

$$P_G(\mathbf{Y}, \mathbf{Z}|\mathbf{S}) = P_G(\mathbf{Y}|\mathbf{S})P_G(\mathbf{Z}|\mathbf{S}). \quad (2)$$

2.2 k-tree and tree-width

Tree-width is a metric on graphs that measures how tree-like a graph is. There are a few alternative definitions for tree-width. The one most relevant to this work is via the notion of k-tree.

Definition 2. Let $k \geq 1$ be an integer. The class of graphs called *k-trees* is defined inductively as follows:

1. A k-tree of exactly k vertices is a k-clique.
2. Given a k-tree $H = (U, F)$ with $n - 1$ vertices ($n > k$), a k-tree $G = (V, E)$ of n vertices is obtained by adding a new vertex $v \notin U$ and connecting it to **exactly k vertices forming a k-clique $C \subseteq U$**, i.e.,

$$V = U \cup \{v\}, \quad E = F \cup \{(v, u) : u \in C\}.$$

When $k = 1$, k-trees are 1-trees, which are simply trees under the regular sense [3]. However, for $k \geq 2$, k-trees are not trees as they contain cycles. We call any subgraph of a k-tree a *partial k-tree*.

Definition 3. Let G be a graph. Then the *tree-width* of G is the smallest number k for which G is a partial k-tree.

The following assertion gives a stronger relationship between graphs of tree-width k and k-trees.

Proposition 1. *For every fixed $k \geq 1$, k-trees are maximal graphs of tree-width k.*

As defined, a k-tree with n vertices is generated through a recursive process that constructs a sequence of (sub)-k-trees whose numbers of vertices are $k, k+1, \ldots, n$. During the generations, vertices and edges are created via rules 1 and 2 in Definition 2. This leads to the following further notes on the notion of k-tree.

First, by Definition 2, every vertex in a k-tree is generated with respect to a certain subset of other vertices. Typically, an application of rule 2 generates a new vertex and associates it with an existing clique of k vertices to form a $(k+1)$-clique. For vertices in the k-clique generated by rule 1, no explicit association of a vertex to others is given. If vertices in the k-clique are considered generated one at a time, like by rule 2, then there is an assumed precedence order in the k-clique such that vertex u precedes vertex v if and only if v is generated with respect to u.

Definition 4. Let $G = (V, E)$ be a k-tree for some $k \geq 1$. An associated *precursor function* for G is a mapping $\pi : V \to 2^V$ that designates a subset of vertices for every vertex v, such that

(1) if $v \in C_0$, the k-clique created with rule 1, then $\pi(v) \subseteq C_0 \setminus \{v\}$; and $\forall u, v \in C_0$, $\pi(u) \subseteq \pi(v)$ if and only if $\pi(v) \not\subseteq \pi(u)$.

(2) if v is generated with rule 2 of Definition 2, then $\pi(v) = C$, where C is given by rule 2, or

Note that case (2) of precursor function π imposes a total order for the vertices in set C_0 generated by rule 1 and hence there exists a vertex, named v_1, such that $\pi(v_1) = \emptyset$. Specifically, vertices in set C_0 are ordered as $(v_1, v_2, \ldots, v_k)$ if and only if $\emptyset = \pi(v_1) \subset \pi(v_2) \subset \cdots \subset \pi(v_k)$ where $v_i \in C_0$, $i = 1, \ldots, k$.

Second, different construction processes of the same k-tree may result in different associated *precursor function*. In this paper, when a k-tree is assumed, some associated precursor function is also assumed.

Third, based on Definition 2, the total number of $(k+1)$-cliques in the k-tree of n vertices, for $n > k$, is exactly $n - k$.

Definition 5. Let $k \geq 1$. A *spanning k-tree* of graph G is a spanning subgraph of G that is a k-tree.

2.3 Markov k-tree:

Definition 6. Let $k \geq 1$ be an integer. A *Markov k-tree* is a Markov network over n random variables $\mathbf{X} = \{X_1, \ldots, X_n\}$ with a topology graph $G = (\mathbf{X}, E)$ being a k-tree. The joint probability distribution function of the Markov k-tree is defined by $P_G(\mathbf{X})$:

$$P_G(\mathbf{X}) = \prod_{X_i \in \mathbf{X}} P(X_i | \pi(X_i)), \tag{3}$$

where π is some associated precursor function[1]. Note that the joint probability $P_G(\mathbf{X})$ given in equation 3 has the right-hand-side that is actually derived using the chain-rule of multivariate probability and conditional independence by Markov properties. Specifically, the right-hand-side is obtained by following the reversed process of constructing the k-tree that corresponds to the associated precursor function π.

[1]Our previous [8] work proves that the joint probability defined for a given Markov k-tree is invariant of its associated precursor function π.

2.4 Mutual Information

Our discussions on approximation of Markov networks are based on Shannon's information theory [43].

Definition 7. Let $\mathbf{X}$ be a finite set of random variables with distribution $P(\mathbf{X})$. Then the *entropy* $H(\mathbf{X})$ of variables $\mathbf{X}$ is defined as $H(\mathbf{X}) = -\sum P(\mathbf{X}) \log_2 P(\mathbf{X})$, where the sum is over all values $\mathbf{x}$ of variable set $\mathbf{X}$.

While the Shannon's entropy accounts for the averaged number of (binary) bits to encode random variable values of one distribution, another notion of entropy can be used to measure the difference between two distributions [31].

Definition 8. Let P and Q be two distributions for the set $\mathbf{X}$ of random variables. Then the *relative entropy* or *Kullback–Leibler divergence* between P and Q is defined as

$$D_{KL}(P \parallel Q) = \sum P(\mathbf{X}) \log_2 \frac{P(\mathbf{X})}{Q(\mathbf{X})}, \tag{4}$$

where the sum is over all values $\mathbf{x}$ of the variable set $\mathbf{X}$.

In particular, with $P(X_i, X_j)$ representing the joint probability between two random variables X_i and X_j and $Q(X_i, X_j) = P(X_i)P(X_j)$ representing their independence distribution, we obtain

Definition 9. Let P be a joint distribution for random variables X_i and X_j. Their *mutual information* is defined as

$$I(X_i; X_j) = \sum P(X_i, X_j) \log_2 \frac{P(X_i, X_j)}{P(X_i)P(X_j)}, \tag{5}$$

where the sum is over all value pairs (x_i, x_j) of the variable pair X_i and Y_j.

Equation 5 can be extended to the following mutual information between a single variable X_i and a set $\mathbf{Y}$ of random variables:

$$I(X_i; \mathbf{Y}) = \sum P(X_i, \mathbf{Y}) \log_2 \frac{P(X_i, \mathbf{Y})}{P(X_i)P(\mathbf{Y})}, \tag{6}$$

where the sum is over all value pairs (x_i, y) of the variable pair X_i and $\mathbf{Y}$.

3 Approximation with k-tree Topology

We now discuss approximate of Markov networks with topologies specified with graphs. Let $\mathbf{X} = \{X_1, \ldots, X_n\}$ be a set of random variables and $P(\mathbf{X})$ be a probabilistic distribution of a Markov network over variables $\mathbf{X}$ with an unknown topology. For graph $G = (V, E)$, with $|V| = n$, we denote with $P_G(\mathbf{X})$ the *distribution of a Markov network over random variables* $\mathbf{X}$ *with topology* $G = (\mathbf{X}, E)$.

Definition 10. Let integer $n \geq 1$ and $\mathcal{G}_n$ be a class of graphs (of n vertices). The *optimal approximation of* $P(\mathbf{X})$ *with topology* $\mathcal{G}_n$ is a distribution $P_{G^*}(\mathbf{X})$ such that

$$G^* = \arg \min_{G \in \mathcal{G}_n} D_{KL}(P(\mathbf{X}) \parallel P_G(\mathbf{X})).$$

Chow and Liu [12] initiated a seminal work on approximation with the topology class $\mathcal{T}_n$ of trees. They proved that for the tree class, the optimal approximation is via a tree topology that yields the maximum sum of mutual information $\sum_{X_i \in \mathbf{X}} I(X_i; X_i')$, where X_i' is the *predecessor* variable of variable X_i. Such a tree can be found by the algorithm that solves the maximum spanning tree problem in loglinear time [40].

Our previous work [8] extended the result to the topology class of k-trees and shows that the optimal approximation can be achieved with a spanning k-tree topology yielding the maximum mutual information

$$\sum_{X_i \in \mathbf{X}} I(X_i; \pi(X_i)) \tag{7}$$

for any associated precursor function π.

The derivation of the result in (7) is based on the connection between the minimization of the KL-divergence and the maximization of a spanning k-tree, as follows. Let G be a spanning k-tree with an associated precursor function π. Applying equation 3 to the KL-divergence formula

$$D_{KL}(P(\mathbf{X}) \parallel P_G(\mathbf{X})) = \sum P(\mathbf{X}) \log_2 \frac{P(\mathbf{X})}{P_G(\mathbf{X})} \tag{8}$$

to yield

$$\begin{aligned} D_{KL}(P(\mathbf{X}) \parallel P_G(\mathbf{X})) &= \sum P(\mathbf{X}) \log_2 \frac{P(\mathbf{X})}{P_G(\mathbf{X})} \\ &= \sum P(\mathbf{X}) \Big(\log_2 P(\mathbf{X}) - \log_2 P_G(\mathbf{X}) \Big) \\ &= \sum P(\mathbf{X}) \log_2 P(\mathbf{X}) - \sum P(\mathbf{X}) \sum_{X_i \in \mathbf{X}} \log_2 P(X_i | \pi(X_i)) \\ &= -H(\mathbf{X}) - \sum P(\mathbf{X}) \sum_{X_i \in \mathbf{X}} \log_2 P(X_i | \pi(X_i)) \end{aligned} \tag{9}$$

where the $\sum P(\mathbf{X})$ is over all values of $\mathbf{X}$ and the last term on the right-hand-side of the last equation in (9) can be rewritten as

$$\begin{aligned} &\sum \sum_{X_i \in \mathbf{X}} P(\mathbf{X}) \log_2 \frac{P(X_i, \pi(X_i))}{P(\pi(X_i))} \\ &= \sum_{X_i \in \mathbf{X}} \sum P(\mathbf{X}) \log_2 \frac{P(X_i, \pi(X_i))}{P(\pi(X_i)) P(X_i)} + \sum_{X_i \in \mathbf{X}} \sum P(\mathbf{X}) \log_2 P(X_i) \\ &= \sum_{X_i \in \mathbf{X}} \sum P(X_i, \pi(X_i)) \log_2 \frac{P(X_i, \pi(X_i))}{P(\pi(X_i)) P(X_i)} + \sum_{X_i \in \mathbf{X}} \sum P(X_i) \log_2 P(X_i) \\ &= \sum_{X_i \in \mathbf{X}} I(X_i; \pi(X_i)) - \underbrace{\sum_{X_i \in \mathbf{X}} H(X_i)}_{\text{Sum of entropies of components } X_i} \end{aligned} \tag{10}$$

where the second last equality holds because $P(\mathbf{X})$ is projected on component $(X_i, \pi(X_i))$ and component X_i with P values over other components summed up to 1.

Because both $H(\mathbf{X})$ in (9) and $\sum_{X_i \in \mathbf{X}} H(X_i)$ in (10) are invariant of the choice of topology G, combining equations (10) with (9) leads to the conclusion that a topology G to minimize distance $D_{KL}(P(\mathbf{X}) \parallel P_G(\mathbf{X}))$ if and only if it maximizes sum of mutual information $\sum_{X_i \in \mathbf{X}} I(X_i; \pi(X_i))$, where π is any associated precursor function with G.

We now turn the latter problem into a graph-theoretic problem. Let $k \geq 1$ be an integer and $[n]$ be the positive integer set.

Definition 11. Let $f : [n]^{k+1} \to \mathbb{R}$ be a function. The problem of the *f-based maximum spanning k-tree*, denoted by MSkT(f), is defined as follows: given an underlying undirected graph $G = (V, E)$, find a maximum spanning k-tree H of G that maximizes the objective function $\sum_{\Delta \in \mathcal{K}_H} f(\Delta)$, where $\mathcal{K}_H$ denotes the set of all $(k+1)$-cliques in H.

Note that, to make the problem MSkT(f) well-defined, the input graph G only needs to include values of the function f over those $(k+1)$-cliques that may appear in some k-tree of G; it is not necessary to provide f for all $(k+1)$-subsets of V. We often omit f from the problem name, writing simply MSkT, when f is understood from the context or need not be explicitly referenced in the discussion. Note that, to make the problem MSkT(f) well defined, the input graph G only needs to specify values of the function f for those $(k+1)$-cliques that may appear in a k-tree of G. It is not necessary to define f for all $(k+1)$-subsets of V.

Theorem 1. *The optimal approximation of Markov networks with k-tree topology can be achieved by solving the problem* MSkT(f)*, where for every $(k+1)$-clique $\Delta = \{X\} \cup \pi(X)$, $f(\Delta) = I(X; \pi(X))$.*

Unfortunately, the intractability of the following problem makes MSkT difficult to solve.

Proposition 2 ([6]). *The following decision problem is NP-hard on every fixed $k \geq 2$: Determining if a given graph G possesses a spanning k-tree as its subgraph.*

4 β-retaining MSkT

We now turn into a restricted version of problem MSkT.

Definition 12. Let β be a class of graphs. *β-retaining* MSkT is the MSkT problem whose outputted maximum spanning k-tree is required to include some spanning subgraph $B \in \beta$ designated in the input graph G.

By the definition, in the graph G as the input for the *β-retaining* MSkT problem, a spanning subgraph B of G, which belongs to the class β, needs to be specified. Our discussion will be focused on β that is specifically the class of bounded-degree spanning trees. That is, problem *β-retaining MSkT* requires its outputted maximum spanning k-tree to retain the designated bounded-degree spanning tree from the input graph G.

Lemma 1. *Let G be a k-tree with $n > k$ vertices. Then*

(1) *Every edge in G belongs to some $(k+1)$-clique in G.*
(2) *During creation of G with Definition 2, no edge (u, v) can be created after u and v have been created. That is, edge (u, v) can only be created when either u or v being created.*
(3) *No $(k+2)$-clique can exist in k-tree G.*

Claim (1) in the above lemma can be justified by Definition 2 that an edge either belongs to the initial k-clique or is one of the k edges connected to an existing k-clique from a newly added vertex. Claim (2) is based on the observation that edge (u, v) can only be created when either u or v is being created, not after they are both created. Claim (3) is true because there would have to be a vertex created the last in a $(k+2)$-clique, were it to exist. When the vertex is being created,

by the rules of k-tree, it can only be connected with k edges to k out of the other $k+1$ vertices in the $(k+2)$-clique. But the edge it shares with the remaining vertices in the $(k+2)$-clique can never be created by Claim (2).

Definition 13. Let G be a k-tree, $k \geq 1$, and Δ be any $(k+1)$-clique in G. Then any $(k+1)$-clique Δ' in G is called a *neighbor of* Δ if $|\Delta \cap \Delta'| = k$.

Definition 14. Let a subset $S \subseteq V$ of vertices in graph $G = (V, E)$. S *separates* the graph into two or more *disconnected components* if S disjoins with all these components and any path connecting vertices between any pair of the disconnected components should contains some vertex in S.

Lemma 2. *Let $G = (V, E)$ be a k-tree and Δ be a $(k+1)$-clique in G. If the removal of Δ disconnects G into m connected components, then Δ has exactly m neighbors in the clique tree of G.*

Proof:

Let Δ be a $(k+1)$-clique in the k-tree G. Denote with $\mathcal{N}(\Delta)$ the set of all neighbors of Δ.

Let $\Delta_1, \Delta_2 \in \mathcal{N}(\Delta)$ be two neighbors of Δ and z_1, z_2 be two vertices such that $z_1 \in \Delta_1 \backslash \Delta$ and $z_2 \in \Delta_2 \backslash \Delta$. It is clear that $z_1 \notin \Delta$ and $z_2 \notin \Delta$. We claim that z_1 and z_2 cannot belong to the same connected component due to the separation of G by the $(k+1)$-clique Δ.

To see the claim is true, consider otherwise there is a path $p = \{(y_0, y_1), (y_1, y_2), \ldots, (y_r, y_{r+1})\}$, for some $r \geq 0$, connecting z_1 and z_2, where $y_0 = z_1$ and $y_{r+1} = z_2$, and the set $V(p)$ of vertices on p is entirely disjoint with Δ. We require that p is the shortest, i.e., consisting of the smallest number of edges.

Assume y_j, for some j, $1 \leq j \leq r$, to be the vertex created the last on the path p. That is, it is created at the same time edges (y_{j-1}, y_j) and (y_j, y_{j+1}) are created. By the rules of k-tree, edge (y_{j-1}, y_{j+1}) exists. Therefore, p is not the shortest, contradicting the assumption.

Without loss of generality, assume that z_1 is the vertex created the last on path p (along with edge (z_1, y_1)). There are two scenarios to consider for this. (a) All vertices in set $\Delta_1 \cap \Delta_2$ already exist before (z_1, y_1) is created. This implies that (z_1, y_1) is created along with $(k+1)$-clique Δ_1, resulting in a $(k+2)$-clique created in G. This contradicts Lemma 1. See Figure 1(a) for an illustration. (b) There is one vertex v in set $\Delta_1 \cap \Delta_2$ does not exist before (z_1, y_1) is created. This implies that $(k+1)$-clique Δ_2 does not exist either. However, when vertex v is created, it should be created along with edges (z_1, v) and (z_2, v), suggesting that set $\Delta_1 \cup \Delta_2$ is a $(k+2)$-clique. This contradicts Lemma 1. See Figure 1(b) for an illustration.

The above argument shows that for any two vertices $z_1 \in \Delta_1$ and $z_2 \in \Delta_2$, where $\Delta_1, \Delta_2 \in \mathcal{N}(\Delta)$ are neighbors of $(k+1)$-clique Δ, z_1 and z_2 belong to two different connected components separated by Δ. So if the number of connected components is m, $|\mathcal{N}(\Delta)| \leq m$. □

Definition 15. A k-tree G has *bounded branches* if every $(k+1)$-clique Δ in G has a bounded number of neighbors. A graph H is *bounded branching-friendly* if every k-tree that contains H as a spanning subgraph has bounded branches.

Theorem 2. *Let β be the class of bounded-degree trees. Then any graph in β is bounded branching friendly.*

Proof: Let $H \in \beta$ be any bounded-degree tree in β. We will show that every k-tree G that contains H as a spanning tree has bounded branches. That is to show that every $(k+1)$-clique in k-tree G separates G into a bounded number of connected components, and then by Lemma 2, every $(k+1)$-clique in such a G has bounded number of neighbors.

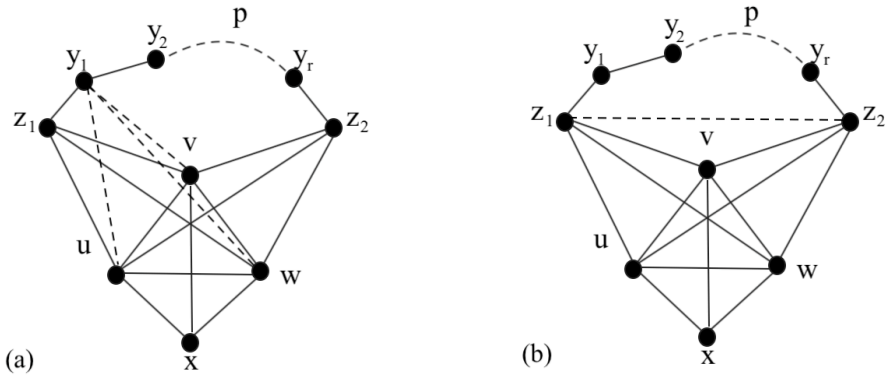

Figure 1: A part of a 3-tree, containing three 4-cliques $\Delta = \{u, v, w, x\}$, $\Delta_1 = \{u, v, w, z_1\}$, and $\Delta_2 = \{u, v, w, z_2\}$, to illustrate the argument in the proof for Lemma 2 that a path disjoint with vertex set Δ but connecting z_1 and z_2 should not exist, where path $p = \{(z_1, y_1), (y_1, y_2), \ldots, (y_r, z_2)\}$ is the assumed path. The proof justifies that z_1 would have to be the last vertex on the path p to be created. Yet this would result in: either (a) a 5-clique $\{u, v, w, z_1, y\}$ is created; or (b) A 5-clique $\{u, v, w, z_1, z_2\}$ is created. Scenarios (a) and (b) both contradict Lemma 1.

Assume constant $d \geq 1$ to be the highest degree of vertices in H and let Δ be any $(k+1)$-clique in the k-tree $G = (V, E)$ that contains H as a spanning tree. Let $S \subseteq V$ be any subset of vertices, with $|S| = j$. We claim that the number of connected components in G separated by S is at most $dj - j + 1$. We prove the claim by induction on j.

When $j = 1$, S contains only one vertex, which can separate H into at most d components. Since H is a spanning tree of G, the vertex can separate G into at most $d = d \times 1 - 1 + 1$ components

Assume S contains $j = m$ vertices and it separates G into at most $dm - m + 1$ connected components. Consider now the case that S contains $j = m + 1$ vertices. Let vertex $v \in S$ be the additional vertex. By assumption, set $S \setminus \{v\}$ should separates G into at most $dm - m + 1$ connected components. Let C be one such component that v belongs to. Then v can further separate subgraph $C \cap H$, thus subgraph $C \cap G$ into at most d components. Together with the other $dm - m$ components, these are $dm - m + d = d(m + 1) - (m + 1) + 1$ connected components separated by S of $m + 1$ vertices.

Apply the proved claim to $(k+1)$-clique Δ, it is clear that Δ separates G into at most $d(k+1) - k$ connected components. Since Δ is an arbitrary $(k+1)$-clique in G, by Lemma 2 and Definition 15, G has bounded branches. And again by Definition 15, we conclude that H in β is bounded branching-friendly. □

Corollary 1. *Let β be the class of Hamiltonian paths. Then any graph in β is bounded branching friendly.*

5 Algorithm

In this section, we show that, for any bounded branching friendly class β of graphs, the β-retaining MSkT problem can be solved in polynomial time for every fixed $k \geq 1$. We will discuss such an algorithm extensively for β that is the class of bounded-degree spanning trees, which can be generalized to any bounded branching friendly class β.

5.1 Tree-Decomposition Representation

We first introduce some further notation to facilitate discussions. Consider again Definition 2 for k-trees. Each k-tree can be viewed as a collection $\mathcal{K}$ of $(k+1)$-cliques. Each $(k+1)$-clique is formed by adding a new vertex v to an existing k-clique C, creating the clique $C \cup v$. However, a single k-clique C may be contained in multiple $(k+1)$-cliques. To clearly describe the relationships among these $(k+1)$-cliques, we label each one uniquely as Δ_v^u, where v is the newly introduced vertex that forms the clique, and $u \neq v$ is a vertex satisfying:

(1) either $u = \epsilon$, indicating that Δ_v^u was the first $(k+1)$-cliques created or
(2) $\exists$ $(k+1)$-clique Δ_u^x, for some $x \in V \cup \{\epsilon\}$ and $u \in V$, which is created earlier, with $|\Delta_u^x \cap \Delta_v^u| = k$.

In (2), Δ_u^x is called the *parent* of Δ_v^u, denoted with $\rho(\Delta_v^u) = \Delta_u^x$. It is clear that every $(k+1)$-clique, except the one first created (that does not have a parent), has a unique parent.

Proposition 3. *The relation defined by parent ρ between $(k+1)$-cliques in k-tree G forms a rooted-tree topology $(\mathcal{K}_G, \mathcal{T}_G)$, where*

(1) $\mathcal{K}_G = \{\Delta : \Delta$ is a $(k+1)$-clique in $G\}$;
(2) $\Delta_x^\epsilon \in \mathcal{K}_G$, for some $x \in V$, is the designated root;
(3) $\mathcal{T}_G = \{(\Delta_u^x, \Delta_v^u) : \Delta_u^x, \Delta_v^u \in \mathcal{K}_G,\ \Delta_u^x = \rho(\Delta_v^u)\}$.

We call $(\mathcal{K}_G, \mathcal{T}_G)$ a *tree-decomposition*[2] of k-tree G. We have the following important property for tree-decomposition.

Proposition 4. *Let Δ' and Δ'' be two $(k+1)$-cliques on a tree-decomposition of k-tree G. Let vertex $v \in \Delta' \cap \Delta''$. Then $v \in \Delta$ for every $(k+1)$-clique Δ on the path between Δ' and Δ'' on the tree.*

Proof: Since Δ is on the path between Δ' and Δ'' on the tree, there are only two scenarios about their creations. (1) Δ is an "ancestor" of both Δ' and Δ''. (2) Δ is a descendent of one and an ancestor of another. In both scenarios, if $v \notin \Delta$, there are duplicated creations of vertex v, contradicting that k-tree definition. □

The notation of tree-decomposition offers a higher level view on a k-tree and makes the discussion easier on algorithms for k-tree optimization. In particular, construction of a k-tree is equivalent to construction of the tree-decomposition of the corresponding k-tree.

Specifically, the tree-decomposition of a k-tree is a tree rooted at $(k+1)$-clique Δ_x^ϵ for some $x \in V$, with tree nodes, both internal and leaf nodes, drawn from set $\mathcal{K}_G$. Since the tree is rooted, it is directional. Therefore, $(k+1)$-clique Δ_v^u is an internal node with a child node Δ_y^v, for any $y \in V$, if and only if $\rho(\Delta_y^v) = \Delta_v^u$. Formally, we need the following technical results in the next section.

[2]The notion of tree-decomposition was first introduced independently from the notion of k-tree [41]. It is used in this section solely for the purpose of discussion our algorithms on k-trees.

Definition 16. *Let $(\mathcal{K}_G, \mathcal{T}_G)$ be a tree-decomposition of a k-tree $G = (V, E)$, with root Δ_x^ϵ, $x \in V$. For any $\Delta \in \mathcal{K}_G$, we define the subset $V_\Delta \subseteq V$ by*

$$V_\Delta = \{v : v \in \Delta',\ \Delta' \in \mathcal{K}_G,\ \Delta' \text{ is a descendant of } \Delta \text{ in } (\mathcal{K}_G, \mathcal{T}_G), \text{ including } \Delta \text{ itself}\}.$$

That is, V_Δ contains all vertices in the $(k+1)$-cliques that are descendants of Δ, **including Δ itself**. Clearly, if Δ is the root of the tree-decomposition, then $V_\Delta = V$.

Proposition 5. *Let $(\mathcal{K}_G, \mathcal{T}_G)$ be a tree-decomposition of k-tree $G = (V, E)$. Then the subtree rooted at $(k+1)$-clique Δ is a tree-decomposition of the induced subgraph of the k-tree G by vertex set V_Δ.*

5.2 Dynamic Programming

By Proposition 5, a tree-decomposition for k-tree G can be constructed by building sub-tree-decompositions and assemble them into the whole tree-decomposition. This leads to the following repetitive (thus recursive) process to construct a tree-decomposition . Specifically, let Δ_v^u be an internal node of the tree with children $\Delta_{w_i}^v$, $i = 1, 2 \ldots, m$, for some $m \geq 1$. Building the subtree rooted at Δ_v^u with the children can be done by first connecting child $\Delta_{w_m}^v$ to Δ_v^u and then continuing to build the subtree rooted at Δ_v^u with the rest of the children $\Delta_{w_i}^v$, $i = 1, 2 \ldots, m-1$.

We now discuss the details. To create the child $\Delta_{w_i}^v$ from parent node Δ_v^u, vertex w_i in G can only be one that was not created before. This means that the creation process needs to keep the record of an exact set from which a new vertex can be drawn for creation of a subtree rooted at Δ_v^u. Because β is a class of bounded branches friendly graphs, by the proofs of Lemma 2 and of Proposition 4, this set of vertices is actually partitioned into at most c exclusive subsets, for some constant $c \geq 1$; vertices from only one of these subsets can be drawn for creation of the subtree. In other words, these subsets correspond to some of the connected components of H separated by the $(k+1)$-clique Δ_v^u. Since H is a known subgraph given in the input graph G, vertices in every one of these subsets are known when Δ_v^u is specified. This suggests that every subset can be retrieved with an index identifier and there are a constant number of such indexes for these exclusive subsets.

Specifically, assume $\lambda \geq 1$ to be the largest number such that H is separated by any $(k+1)$-clique into at most λ connected components. Let $[\lambda]$ denote the set $\{1, 2, \ldots, \lambda]$. Given tree node (i.e., $(k+1)$-clique) Δ_v^u, let $S : [\lambda] \to 2^V$ denote the function that maps any identifier $i \in [\lambda]$ to the set of vertices in the i^{th} connected component (resulted from the separation by Δ_v^u). Then the set of vertices in the subtree rooted at Δ_v^u, excluding those in Δ_v^u, can be written as $V_{u,v,\mathcal{I}} = \bigcup_{i \in \mathcal{I}} S(i)$, for some subset $\mathcal{I} \subseteq [\lambda]$. We will show that that set $V_{u,v,\mathcal{I}}$ is uniquely determined by $\mathcal{I}$ along with Δ_v^u during the tree construction process.

Definition 17. Let f be a real-value function with the argument being a $(k+1)$-clique. We define real-value function $F(\Delta_v^u, \mathcal{I})$ to be the maximum sum of f values over all $(k+1)$-cliques (*excluding* Δ_v^u) in a subtree rooted at Δ_v^u that contains vertices in $V_{u,v,\mathcal{I}} \cup \Delta_v^u$.

Then we can derive the following recurrence for function F:

$$F(\Delta_v^u, \mathcal{I}) = \max_{i \in \mathcal{I},\, w \in S(i) \setminus \Delta_v^u} \left\{ F(\Delta_w^v, \mathcal{J}_i) + F(\Delta_v^u, \mathcal{I} \setminus \{i\}) + f(\Delta_w^v) \right\} \tag{11}$$

where $\mathcal{J}_i \subseteq [\lambda]$ is a subset of identifiers for the connected components of H separated by Δ_w^v. The recurrence (11) has the base cases $F(\Delta_v^u, \emptyset) = 0$, for all $u, v \in V$.

Lemma 3. *Recurrence (11) together with its base cases computes correctly function* $F(\Delta_v^u, \mathcal{I})$.

Proof: We prove the claim by showing that the recurrence computes correctly the sum of f scores on all $(k+1)$-cliques in the subtree rooted at Δ_v^u, excluding Δ_v^u itself. We do this by induction on h, the number of $(k+1)$-cliques in the subtree rooted at Δ_v^u.

When $h = 1$, the subtree contains only $(k+1)$-clique Δ_v^u itself. This corresponds to the case case $F(\Delta_v^u, \emptyset)$ which has value 0.

We hypothesize that when $h \leq N$, the claim is correct. Consider the case $h = N + 1$. Note that node Δ_v^u, the root of the subtree, has $|\mathcal{I}|$ children nodes, each of which is the root of the subtree containing exactly the vertices in one of the connected components separated by Δ_v^u. The recurrence first adds term $f(\Delta_w^v)$ into the sum, where Δ_w^v is one of the children node and the root of the subtree covering the i^{th} connected component. Term $F(\Delta_w^v, \mathcal{J}_i)$ recursively computes the sum of f scores on the rest of $(k+1)$-cliques in the subtree covering the i^{th} connected component. Term $F(\Delta_v^u, \mathcal{I} \setminus \{i\})$ recursively computes the sum of f scores on the rest of subtrees rooted at Δ_v^u. In both cases, the numbers of $(k+1)$-cliques to compute are at most N and they are both correct by the hypothesis.

Now we justify that given $(k+1)$-clique Δ_v^u and identifiers $\mathcal{I}$, the information of vertices belonging to connected components (of H separated by Δ_v^u) can be determined. Consider vertices in Δ_v^u are identified on H, upon which a search process (e.g., depth-first search) on H can determine connected components. Vertices are associated with these components and each component is associated with an identifier in $\mathcal{I} \subseteq [\lambda]$. The component that vertex u belongs, which can be determined, should be in set $\mathcal{I}$. The same argument applies to $\mathcal{J}_i$. □

Since Recurrence (11) maximizes the sum of f scores on all $(k+1)$-cliques of the subtree rooted at a $(k+1)$-clique except the root, the answer to the β-retaining MSkT(f) problem can be expressed as:

$$\arg \max_{\Delta_u^\epsilon \in V^{k+1}, \mathcal{I} \subseteq [\lambda]} \left\{ F(\Delta_u^\epsilon, \mathcal{I}) + f(\Delta_u^\epsilon) \right\} \tag{12}$$

by examining all possible roots Δ_u^ϵ for a maximum spanning k-tree.

Function F can be implemented with a dynamic programming algorithm based on the recurrence equation (11). This essentially involves establishing a look-up table that computes values of function F on all necessary combinations of $(\Delta_v^u, \mathcal{I})$. For graph of n vertices, the table size is $O(n^{k+1} \times 2^\lambda)$, where λ is the number of components separated by a $(k+1)$-clique and a function in k. For fixed k, λ is a constant. Therefore, each entry in the table can be computed in time $O(n)$ for selecting vertex $w \in S(i)$. This gives rise to the time complexity $O(n^{k+2})$.

Theorem 3. *Let β be the class of all bounded branches friendly graphs. The β-retaining MSkT(f) problem can be solved in time $O(n^{k+2})$, for every fixed $k \geq 1$.*

Theorem 4. *Given any Markov network of n random variables, its k-tree topology approximation with the minimum loss of information can be computed in time $O(n^{k+2})$, for every fixed $k \geq 1$, provided that the found topology retains some designated spanning graph that is bounded branches friendly.*

Corollary 2. *There are polynomial time algorithms for the minimum loss of information approximation of Markov networks with a k-tree topology that retains a designated Hamiltonian path in the input network.*

Corollary 3. *There are polynomial time algorithms for the minimum loss of information approximation of Markov networks with a k-tree topology that retains a designated bounded-degree spanning tree in the input network.*

5.3 Optimality in Complexity

An early study [13] on the β-retaining MSkT problem showed that, for the class β of Hamiltonian paths, the algorithm can be tweaked such that the time complexity can be improved to $O(n^{k+1})$. We believe the technique and the time bound can be generalized to all bounded branches friendly classes β. We omit such a proof which is rather lengthy.

However, the polynomial time $O(n^{k+1})$ for β-retaining MSkT may not be further improved in term of its exponent parameter k. In the following we show strong evidence that this claim is true. For this we defined the following decision problem related to β-retaining MSkT(f) with β being the class of Hamiltonian paths. We assume f is real-value function with argument being a $(k+1)$-clique together with weights of edges they share.

H-MSkT(f):

Input: edge-weighted graph G with an Hamiltonian path H, integer $k \geq 1$, real number S;

Output: "Yes" if and only if there is a spanning k-tree retaining H whose sum of f scores over all $(k+1)$-cliques is at least S.

We now connect problem H-MSkT(f) to the classical graph-theoretic problem k-Clique, which determines if the input graph has a clique of size as the given threshold k. While k-Clique problem can be solved trivially in time $O(n^k)$ on graph of n vertices, any substantial improvement to the upper bound has proved extremely difficult. In particular, it is unlikely [1, 32] to solve problem k-Clique in time $O(n^{k-\epsilon})$ for any $\epsilon > 0$.

Actually the complexity of the k-Clique problem has been more systematically investigated within the realm of parameterized complexity theory [14]. It studies computational complexity of algorithms in terms of both a special parameter k in the input and the input size n. Specifically, a problem is *fixed parameter tractable* if it admits algorithms of time complexity of $f(k)n^{O(1)}$ for some recursive function f. Such problems thus belong to the class FPT. The framework designates all problems solvable in time $n^{f(k)}$ to the class XP, which contains the FPT as a subclass. Problem k-Clique belongs to XP but it is unlikely to be in the class FPT. In particular, it is complete for subclass $W[1]$ of XP . Thus solving k-Clique in time $t(k)n^{O(1)}$, for some function $t(k)$ independent of n would lead to an unlikely breakthrough in computational complexity theory.

We now show such difficulties for k-Clique also passes on to H-MSkT(f) and thus to the problem β-retaining MSkT(f) for all bounded branches friendly classes β. This complexity connection from k-Clique is through a transformation from k-Clique to H-MSkT(f), which is a "parameterized reduction" in the sense that the parameter k in k-Clique is directly translated to the parameter k in H-MSkT(f), independent of the input graph size n.

Proposition 6. *Problem k-Clique can be transformed to problem H-MSkT(f) via a parameterized reduction.*

Proof: We sketch the desired reduction. Given an input instance of k-Clique consisting of an undirected graph $G = (V, E)$ and parameter k, we construct an instance of H-MSkT(f) that includes an edge-weighted graph G', a Hamiltonian path H in G', parameter k', and a score threshold σ. Specifically,

(1) Graph $G' = (V', E')$, where $V' = \{l(x_i) : x_i \in V\}$ and $l : V \to [n]$ is a one-to-one mapping with $n = |V|$. Further, $E' = V' \times V'$, i.e., G' is a complete vertex-labeled graph.

(2) Edge weights are defined by a function $w : E' \to \mathbb{R}$ such that for every edge $(l(u), l(v)) \in E'$,

$$w(l(u), l(v)) = \begin{cases} 1 & (u, v) \in E, \\ 0 & (u, v) \notin E. \end{cases}$$

(3) The Hamiltonian path in G' is

$$H = \{(i, i+1) : (i, i+1) \in E',\ 1 \leq i < n\}.$$

(4) Parameter $k' = k - 1$.
(5) Score threshold $\sigma = 1$.

We define a real-valued function f on $(k'+1)$-subsets of V'. The values of f are specified implicitly by the following expression: for any subset $\Delta \subseteq V'$ with $|\Delta| = k' + 1$,

$$f(\Delta) = \prod_{x,y \in \Delta} w(x, y).$$

Thus $f(\Delta) = 1$ if and only if all edges among the vertices in Δ have weight 1, and otherwise $f(\Delta) = 0$.

Now suppose G contains a clique of size k, say $\{x_1, x_2, \ldots, x_k\}$. Then the corresponding set of vertices in G', $\Delta = \{l(x_1), l(x_2), \ldots, l(x_k)\}$, forms a $(k'+1)$-clique with $f(\Delta) = 1$. Since G' is complete, there exists a spanning k'-tree containing H that includes this $(k'+1)$-clique, and the sum of f-scores in the tree is at least $\sigma = 1$.

Conversely, suppose G' has a spanning k'-tree containing H whose total f-score is at least σ. Then at least one $(k'+1)$-clique $\Delta = \{l(x_1), l(x_2), \ldots, l(x_k)\}$ in the k'-tree must satisfy $f(\Delta) = 1$. By the definition of w, all corresponding edges (x_i, x_j) must belong to E, implying that $\{x_1, x_2, \ldots, x_k\}$ forms a clique in G. Hence G contains a clique of size k.

We note that the values of f are defined implicitly through the above expression rather than being explicitly enumerated for all $(k'+1)$-subsets. A stronger reduction could explicitly list the values of f for each subset in the input. However, the implicit representation suffices to establish the parameterized reduction. □

Proposition 6 shows that the algorithm presented earlier is unlikely to be significantly improved for the β-retaining MSkT(f) problem and for the optimal approximation of Markov networks with k-tree topology, assuming standard complexity-theoretic conjectures.

6 Conclusions

We have demonstrated polynomial-time algorithms to solve the β-retaining maximum spanning k-tree problem, for every fixed integer $k \geq 1$. Our research has revealed that a maximum spanning k-tree topology corresponds to an optimal approximation of Markov networks (with the minimum information loss). While finding a maximum spanning k-tree is computationally intractable, we have shown that the problem can be solve efficiently when the desired k-tree also retains a designated spanning subgraph in graph classes of certain characteristics, namely being bounded branches friendly. We also demonstrated strong evidence that our algorithms are likely the optimal in time complexity.

In this paper, we have considered two classes β of graphs: bounded-degree spanning trees and its subclass of Hamiltonian paths. The MSkT problems that retains graphs from these two classes have arisen from practical applications, for example, a biomolecule 3D structure graph containing the backbone as a designated Hamiltonian path [8] and a gene or metabolic network containing a known, critical pathway as a designated spanning tree [9]. As the maximum spanning k-tree problem is essential to network approximation, it is of interest to investigate larger classes of graphs that can be retained in finding maximum spanning k-tree. One such class may be 2-connected graphs [42] that lie between the class of trees and and class of 2-trees.

Our work echoes well previous research in parameterized algorithms for Bayesian network learning. Structure learning of Bayesian networks is generally NP-hard [10] and also intractable even when the structure topology is restricted to tree-width 2 [27,38]. While heuristic algorithms have been resorted to [15], exact algorithms to learn Bayesian networks under other constraints have also been investigated. Typically, Bayesian network with vertex cover number bounded by k was proposed as an alternative to bounded tree-width networks and a time-$O(4^k n^{O(k)})$ algorithm has been discovered [28]. Yet at the same time the problem was also shown to be $W[1]$-hard, excluding the hope to admit a polynomial-time algorithm with degree independent of the parameter k. More recently, the research has been extended to other structural constraints [19]. Typically, it has been proved that learning optimal networks where the network or its moralized graph have maximum degree 2 or connected components of size at most c, $c \geq 3$, is NP-hard. The same research also shows that learning an optimal network with at most k edges in the moralized graph presumably is paramaterized intractable while an optimal network with at most k arcs is parameterize tractable. Our present work may offer insights into learning of Bayesian networks of tree-width k that may retain some subgraph of a desired structural property.

References

[1] A. Abboud, A. Backurs, and V. V. Williams. If the current clique algorithms are optimal, so is valiant's parser. *SIAM Journal on Computing*, 47(6), 2018. doi:10.1137/15M1050705.

[2] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational inference: A review for statisticians. *Journal of the American Statistical Association*, 112(518):859–877, 2017. doi:10.1080/01621459.2017.1285773.

[3] H. L. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theoretical Computer Science*, 209(1):1–45, 1998. doi:10.1016/S0304-3975(97)00228-4.

[4] E. Boix-Adserà, G. Bresler, and F. Koehler. Chow-Liu++: Optimal prediction-centric learning of tree ising models. In *IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 417–426, 2022. doi:10.1109/FOCS52979.2021.00049.

[5] G. Bresler. Efficiently learning ising models on arbitrary graphs. In *Proceedings of the Forty-Seventh Annual ACM Symposium on Theory of Computing (STOC)*, pages 771–782, 2015. doi:10.1145/2746539.2746603.

[6] L. Cai and F. Maffray. On the spanning k-tree problem. *Discrete Applied Mathematics*, 44(1–3):139–156, 1993. doi:10.1016/0166-218X(93)90228-G.

[7] C. Chan and T. Liu. Clustering by multivariate mutual information under chow-liu tree approximation. In *2015 53rd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 993–999, 2015. doi:10.1109/ALLERTON.2015.7447116.

[8] D. Chang, L. Ding, R. Malmberg, D. Robinson, M. Wicker, H. Yan, A. Martinez, and L. Cai. Optimal learning of markov k-tree topology. *Journal of Computational Mathematics and Data Science*, 4:100046, 2022. doi:10.1016/j.jcmds.2022.100046.

[9] P. Chatterjee and N. R. Pal. Discovery of synergistic genetic network: A minimum spanning tree-based approach. *Journal of Bioinformatics and Computational Biology*, 14(1), 2016. doi:10.1142/S0219720016500053.

[10] D. M. Chickering. Learning bayesian networks is np-complete. In *Learning from Data: Artificial Intelligence and Statistics V*, pages 121–130, 1996. doi:10.1007/978-1-4612-2404-4_12.

[11] H. Chitsaz, R. Backofen, and S. C. Sahinalp. biRNA: Fast RNA-RNA binding sites prediction. In *Proceedings of the 9th International Workshop on Algorithms in Bioinformatics (WABI 2009)*, pages 25–36. Springer, 2009. doi:10.1007/978-3-642-04241-6_3.

[12] C. Chow and C. Liu. Approximating discrete probability distributions with dependence trees. *IEEE transactions on Information Theory*, 14(3):462–467, 1968. doi:10.1109/TIT.1968.1054142.

[13] L. Ding. *Maximum Spanning k-Trees: Models and Applications*. PhD thesis, University of Georgia, 2016.

[14] R. Downey and M. Fellows. *Parameterized Complexity*. Springer, 1999.

[15] G. Elidan and S. Gould. Learning bounded treewidth bayesian networks. In *Proceedings of the Twenty-Second Annual Conference on Neural Information Processing Systems*, pages 417–424, 2008.

[16] I. Foster and C. Kesselman. *The Grid: Blueprint for a New Computing Infrastructure*. Morgan Kaufmann, 2003.

[17] A. Gionis, P. Indyk, and R. Motwani. Similarity search in high dimensions via hashing. In *Proceedings of the 25th International Conference on Very Large Data Bases (VLDB)*, pages 518–529, 1999. doi:10.5555/645925.671516.

[18] G. R. Grimmett. A theorem about random fields. *Bulletin of the London Mathematical Society*, 5(1):81–84, 1973. doi:10.1112/blms/5.1.81.

[19] N. Grüttemeier and C. Komusiewicz. Learning bayesian networks under sparsity constraints: a parameterized complexity analysis. *Journal of Artificial Intelligence Research*, 74:1225–1267, 2022. doi:10.1613/JAIR.1.13138.

[20] C. Gunduz, B. Yener, and S. H. Gultekin. The cell graphs of cancer. *Bioinformatics*, 20(suppl_1):i145–i151, 2004. doi:10.1093/bioinformatics/bth915.

[21] N. Hammami and M. Bedda. Improved tree model for Arabic speech recognition. In *Proceedings of the 3rd International Conference on Computer Science and Information Technology (ICCSIT)*, volume 5, pages 521–526, 2010. doi:10.1109/ICCSIT.2010.5563892.

[22] K. Huang, I. King, and M. R. Lyu. Constructing a large node chow-liu tree based on frequent itemsets. In *Proceedings of the 9th International Conference on Neural Information Processing (ICONIP)*, volume 1, pages 498–502, 2002. doi:10.1109/ICONIP.2002.1202220.

[23] M. D. Humphries and K. Gurney. Network 'small-world-ness': a quantitative method for determining canonical network equivalence. *PLOS ONE*, 3(4):e0002051, 2008. doi:10.1371/journal.pone.0002051.

[24] J. Jiao, Y. Han, and T. Weissman. Beyond maximum likelihood: Boosting the chow-liu algorithm for large alphabets. In *The 50th Asilomar Conference on Signals, Systems and Computers*, pages 321–325, 2016. doi:10.1109/ACSSC.2016.7869051.

[25] D. R. Karger and N. Srebro. Learning markov networks: Maximum bounded tree-width graphs. In *Proceedings of the Twelfth Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 392–401, 2001. doi:10.5555/365411.365457.

[26] B. M. King and B. Tidor. MIST: Maximum information spanning trees for dimension reduction of biological data sets. *Bioinformatics*, 25(9):1165–1172, 2009. doi:10.1093/bioinformatics/btp161.

[27] J. H. Korhonen and P. Parviainen. Learning bounded tree-width bayesian networks. In *Proceedings of In 16th International Conference on Artificial Intelligence and Statistics*, 2013.

[28] J. H. Korhonen and P. Parviainen. Tractable bayesian network structure learning with bounded vertex cover number. In *In Proceedings of the Twenty-Eighth Annual Conference on Neural Information Processing Systems*, page 622–630, 2015.

[29] E. Kovács and T. Szántai. On the approximation of a discrete multivariate probability distribution using the new concept of t-cherry junction tree. In *Coping with Uncertainty: Robust Solutions*, pages 39–56. Springer, 2009. doi:10.1007/978-3-540-87837-0_3.

[30] F. R. Kschischang, B. J. Frey, and H. A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, 2001. doi:10.1109/18.910572.

[31] S. Kullback and R. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22(1):79–86, 1951. doi:10.1214/aoms/1177729694.

[32] L. Lee. Fast context-free grammar parsing requires fast boolean matrix multiplication. *Journal of ACM*, 49(1):1–15, 2002. doi:10.1145/505241.505242.

[33] M. Melucci. A brief survey on probability distribution approximation. *Computer Science Review*, 33:91–97, 2019. doi:10.1016/j.cosrev.2019.06.001.

[34] T. Mohseni Ahooyi, J. E. Arbogast, and M. Soroush. Rolling pin method: Efficient general method of joint probability modeling. *Industrial & Engineering Chemistry Research*, 53(52):20191–20203, 2014. doi:10.1021/ie503265u.

[35] L. V. Montiel and J. E. Bickel. Approximating joint probability distributions given partial information. *Decision Analysis*, 10(1):26–41, 2013. doi:10.1287/deca.1120.0253.

[36] K. P. Murphy. *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012.

[37] M. E. J. Newman. The structure and function of complex networks. *SIAM Review*, 45(2):167–256, 2003. doi:10.1137/S003614450342480.

[38] S. Ordyniak and S. Szeider. Parameterized complexity results for exact bayesian network structure learning. *Journal of Artificial Intelligence Research*, 46:263–302, 2013. doi:10.1613/jair.3942.

[39] H. Patil. On the structure of k-trees. *Journal of Combinatorics, Information and System Sciences*, 11(2–4):57–64, 1986.

[40] R. C. Prim. Shortest connection networks and some generalizations. *The Bell System Technical Journal*, 36(6):1389–1401, 1957. doi:10.1002/j.1538-7305.1957.tb01515.x.

[41] N. Robertson and P. D. Seymour. Graph minors. i. excluding a forest. *Journal of Combinatorial Theory, Series B*, 35(1):39–61, 1983. doi:10.1016/0095-8956(83)90079-5.

[42] A. Schrijver. *Combinatorial Optimization*. Springer, 2003. doi:10.1007/978-3-540-68279-0.

[43] C. E. Shannon. A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423, 1948. doi:10.1002/j.1538-7305.1948.tb01338.x.

[44] A. Singh and M. D. Humphries. Finding communities in sparse networks. *Scientific Reports*, 5:8828, 2015. doi:10.1038/srep08828.

[45] N. Srebro. Maximum likelihood bounded tree-width markov networks. *Artificial Intelligence*, 143:123–138, 2003. doi:10.1016/S0004-3702(02)00360-0.

[46] A. Steimer, F. Zubler, and K. Schindler. Chow–Liu trees are sufficient predictive models for reproducing key features of functional networks of periictal eeg time-series. *NeuroImage*, 118:520–537, 2015. doi:10.1016/j.neuroimage.2015.05.089.

[47] J. Suzuki. A novel Chow–Liu algorithm and its application to gene differential analysis. *International Journal of Approximate Reasoning*, 80:1–18, 2017. doi:10.1016/j.ijar.2016.08.001.

[48] T. Szántai and E. Kovács. Hypergraphs as a mean of discovering the dependence structure of a discrete multivariate probability distribution. *Annals of Operations Research*, 193:71–90, 2012. doi:10.1007/s10479-010-0814-y.

[49] V. Y. F. Tan, S. Sanghavi, J. W. Fisher, and A. S. Willsky. Learning graphical models for hypothesis testing and classification. *IEEE Transactions on Signal Processing*, 58(11):5481–5495, 2010. doi:10.1109/TSP.2010.2042478.

[50] M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1–2):1–305, 2008. doi:10.1561/2200000001.

[51] X. Zhang, D. Chang, W. Qi, and Z. Zhan. Tree-like dimensionality reduction for cancer informatics. In *IOP Conference Series: Materials Science and Engineering*, volume 490, page 042028. IOP Publishing, 2019. doi:10.1088/1757-899X/490/4/042028.