# FROM FIELDS TO RANDOM TREES

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

This study introduces a novel method for performing Maximum A Posteriori (MAP) estimation on Markov Random Fields (MRFs) that are defined on locally and sparsely connected graphs, broadly existing in real-world applications. We address this long-standing challenge by sampling uniform random spanning trees (SPT) from the associated graph. Such a sampling procedure effectively breaks the cycles and decomposes the original MAP inference problem into overlapping sub-problems on trees, which can be solved exactly and efficiently. We demonstrate the effectiveness of our approach on various types of graphical models, including grids, cellular/cell networks, and Erdős–Rényi graphs. Our algorithm outperforms various baselines on synthetic, UAI inference competition, and real-world PCI problems, specifically in cases involving locally and sparsely connected graphs. Furthermore, our method achieves comparable results to these methods in other scenarios. The code of our model can be accessed at `https://anonymous.4open.science/r/From-fields-to-trees-iclr-EB75`.

## 1 INTRODUCTION

In this paper, we investigate a novel approach to infer on Markov Random Fields defined over *sparsely and locally connected graphs* via random spanning tree sampling. Formally, an MRF is defined over an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V}$ represents the index set of random variables and $\mathcal{E}$ corresponds to the edge set implying dependencies among these variables. Each random variable in $x_i \in \mathcal{X}$ takes a value from a finite alphabet $\mathcal{X}$ ($x_i$ can be a vector) with node index $i \in \mathcal{V}$, and the joint distribution is expressed as a product of potential functions, each associated with a subset of variables that forms a clique in $\mathcal{G}$. In this paper, we consider minimizing energy functions with unary terms $\theta_i$ and pairwise terms $\theta_{ij}$, then the MAP estimate is:

$$\min_X E(X) = \min_{\{x_i\},\forall i \in \mathcal{V}} \left\{ \sum_{i \in \mathcal{V}} \theta_i(x_i) + \sum_{(i,j) \in \mathcal{E}} \theta_{ij}(x_i, x_j) \right\}. \tag{1}$$

where $X = \{x_i\}_{i \in \mathcal{V}}$. Optimization problem over MRFs expressed in equation 1 is a prevalent model in the realm of probabilistic graphical modeling (Clifford & Hammersley, 1971). Notably, MRF and its counterparts enable the principled and simplified representation of joint probability distributions over a set of variables, conditioned upon the structure of an undirected graph. The intrinsic capability of MRFs to model contextual relationships in data offers a cohesive and efficient framework for addressing inference and estimation challenges across a diverse array of scientific and engineering disciplines, including computer vision (Wang et al., 2013; Su et al., 2021), 5G networks (Kumar et al., 2022), pathology image analysis (Li et al., 2020) and in other fields combining the power of GNN (Xu et al., 2021a; Wu et al., 2020).

The problem in equation 1 is known to be NP-hard in general. Early trials to efficiently and suboptimally solve equation 1 dates back to the 80's of the previous century, when Judea Pearl developed the prestigious Belief Propagation (BP) algorithm (Pearl, 1988). Motivated by this breakthrough, a series of BP variants were further proposed (Wiegerinck & Heskes, 2002; Yedidia et al., 2005; Montanari & Rizzo, 2005). Other prominent methods include Mean Field Approximation (Saito et al., 2012; Zhang, 1993; Zhang & Hanauer, 1995), Graph Cuts (Greig et al., 1989), and Junction Trees (Aji & McEliece, 2000), to name a few. These methods offer varying trade-offs between accuracy and computational efficiency. To the best of our knowledge, no single method stands out as state-of-the-art over all existing problems spanning various scales, topologies, and problem distribution.

In this study, we put our special focus on sparsely and locally connected graphs. In realistic scenarios, such topologies broadly exist in power grid (Cuffe & Keane, 2017), 5G networks (Agiwal et al., 2016; Liu & Zou, 2020), transportation networks (Yunfei Ma & Razavi, 2022), and even social networks (Majeed et al., 2020).

Efficient MAP inference in the corresponding applications is crucial. Beyond our focus, we emphasize that our study can be flexibly extended into other types of graphs.

Diverging from existing well-established strategies, our approach leverages the structure of random spanning trees to efficiently infer on MRFs. In a nutshell, our approach consists of “sampling random trees – solving MRFs on trees – merging” steps. Concretely, instead of solving the MRF on the entire graph directly, we sample multiple spanning trees from the original graph and solve the MRF independently on each tree, on which exact inference is tractable. The final solution to equation 1 is then approximated by merging the solutions from all sampled trees. Our approach thus combines the benefits of exact tree-based inference procedures with the flexibility of sampling methods, creating a balance between computational efficiency and accuracy.

We conduct experiments on synthetic instances, UAI competition instances (uai), and real-world Physical Cell Identity (PCI) instances, covering a variety of problem scales, types of topologies, and energy functions. We observe superior performance against several baselines in the sparse and local network setting, and comparable performance in other settings, which strongly pose the promise of the proposed method in various scenarios.

In summary, our contribution through this proposed method introduces a scalable and efficient sampling-based approach to infer MRFs on locally and sparsely connected graphs, mitigating the drawbacks of existing methods. Superior performance is observed in benchmarks across a wide spectrum. By transcending the traditional methodologies and introducing structural simplification via topological sampling over trees, this work posits rich potential to solve intricate MRF problems in the real world.

## 2 Related Works

In MRFs, the energy function is linked to a graph-structured probability distribution. A significant inference challenge in MRFs is determining the MAP configuration. Although minimizing the energy function of MRF models is NP-hard, advances in inference techniques have significantly expanded the model’s capabilities. The success in solving the MAP estimation problem on cycle-free graphs is highly dependent on the graph’s structure. In these graphs, the MAP problem can be effectively tackled using a variant of the min-sum algorithm (Clifford & Hammersley, 1971; Besag, 1974; Kumar et al., 2005), which facilitates message passing between nodes and serves as an extension of the Viterbi algorithm (Yedidia et al., 2003) to arbitrary cycle-free graphs. For graphs containing cycles, graph cut methods (Komodakis et al., 2007; Roy & Cox, 1998; Boykov et al., 1998; Ishikawa & Geiger, 1998; Szummer et al., 2008; Ishikawa, 2003; Schlesinger & FLACH, 2006) offer a potent solution by employing min-cut/max-flow strategies to efficiently reduce discrete MRFs’ energy.

The belief propagation (BP) algorithm, introduced by Pearl (Pearl, 1982; 1988) in 1982, is an efficient iterative inference algorithm for Bayesian belief networks, functioning through fixed-point message passing. Its adaptability has made it a widespread solution for various types of MRFs. Nonetheless, BP encounters difficulties with models containing loops. Loopy belief propagation (LBP) attempts to resolve this by iterating message passing within graphs, even with loop presence (Weiss & Freeman, 2001; Felzenszwalb & Huttenlocher, 2004; Frey & Mackay, 2002). While LBP has shown efficacy in several vision tasks (Sun et al., 2002), it does not ensure fixed-point convergence, and its theoretical underpinnings remain elusive. The quest for a flexible, convergence-guaranteed method persists, yet significant advancements have been made to enhance BP’s performance. The method proposed by Grim & Felzenszwalb (2023) enhances BP by adjusting the significance of input messages through a discount factor for remote nodes in the message passing chain. Additionally, leveraging graph topology for decomposition mitigates the impact of loops. In Yan et al. (2023), BP’s inefficiency in large-scale MRFs is addressed by constructing a hierarchical framework, facilitating inference via energy connections between layers. Another strategy, detailed by Hamze & de Freitas (2004), involves partitioning graphs into two segments to serve as mutual evidence for updates, although its applicability is limited to graphs with predefined structures. In Kirkley et al. (2021), they propose graph decomposition using primary circles of a specified length from any given node, aiming to circumvent short loop influences, yet it is not effective for large-scale graphs. Integrating tree structures to break loops within the graph, the Junction Tree Algorithm (JTA) (Aji & McEliece, 2000), an exact inference method for arbitrary graphs, entails finding a maximum spanning tree across the largest cliques of a triangulated graph, a task known to be NP-hard, thereby limiting its practicality. In the realm of pairwise MRFs, problems are formulated as integer linear programming (ILP) (Wainwright et al., 2005; Kolmogorov, 2006), where solutions are derived from a dual problem using a convex combination of trees. This class of algorithms, known as tree-reweighted message passing (TRW) techniques, encompasses edge-based (TRW-E) and tree-based (TRW-T) schemes, both of which lack guaranteed convergence, potentially looping infinitely. The sequential TRW-S scheme (Kolmogorov, 2006) achieves a state of weak tree agreement (WTA), ensuring the lower bound stabilizes, albeit requiring substantial time to reach this stage. TRBP has long been considered the state-of-the-art (SOTA) methodology and has been adapted in Xu et al. (2021b) to exploit modern GPUs for accelerated inference processes.

Several approaches have leveraged tree structures to address graph-related challenges. Batra et al. (2010) proposes a graph decomposition method using outer-planar graphs. However, this approach is restricted to planar MRFs (such as grids or superpixel adjacency graphs), and the computational complexity in determining both the required number and structure of subgraphs limits its applicability to general and large-scale graphs. The method in Pletscher et al. (2009) employs tree log-likelihood for approximation, while Skurikhin (2014) combines log-likelihood maximization with gradient descent for inference, specifically on grid structures. Both methodologies rely on voting mechanisms for final predictions. Using trees and Non-parametric Belief Propagation (Savic & Zazo, 2010) improve the accuracy in solving the localization problem in communication networks under 100 nodes. None of these three methods implement message correction on edges, potentially leading to deviations from original messages, as demonstrated in Lemma 1. While Bradley & Guestrin (2010) addresses structure learning, our work specifically focuses on MAP inference in MRFs and CRFs. Additionally, their approach is limited to graphs exhibiting tree-like structures. Trees could also work on binary inference problems (Cesa-Bianchi et al., 2010) through graph transformation, converting tree structures into line graphs for prediction purposes. Our work proposes a novel methodology that addresses two critical challenges in MRFs: circumventing the computational complexities associated with loop structures while maintaining faithful approximations of the original problem formulation. This approach demonstrates particular efficacy in handling large-scale MRF instances and problems that can be formulated within the MRF framework.

## 3 Preliminaries

**Markov Random Field**. MRFs can be used to model probabilistic undirected graphs. We follow the notations in Section. 1 and further suppose $|\mathcal{E}| = M$ and $|\mathcal{V}| = N$. In this paper, each node $i$ corresponds to a discrete state variable $x_i \in \mathcal{X}, \forall i \in \mathcal{V}$, where $\mathcal{X}$ is a finite alphabet. And there is a conditional independence between variables

$$\mathbb{P}(x_i|X\backslash\{x_i\}) = \mathbb{P}(x_i|\{x_j\} \text{ for } (i,j) \in \mathcal{E}). \tag{2}$$

where $\mathbb{P}(\cdot)$ is the probability function throughout this paper. Thus, the distribution on the graph $\mathcal{G}$ can be factorized into a product of local Markov potentials

$$\mathbb{P}(X) = \tfrac{1}{Z}\exp(-E(X)) = \tfrac{1}{Z}\exp\left(-\sum_{i\in\mathcal{V}}\theta_i(x_i) - \sum_{(i,j)\in\mathcal{E}}\theta_{ij}(x_i,x_j)\right) \tag{3}$$

where $Z$ is the partition function, $\theta_i(\cdot)$ denotes the unary energies, $\theta_{ij}(\cdot)$ represent the pairwise interaction energies, and $E$ is the energy function. The specific forms of energy functions are determined by the nature of the problems and they are all known in most cases. In our study, we aim to maximize the posterior probability by finding the optimal configuration of the hidden variables $X := \{x_i\}$ (assign appropriate values to each of the random variables). This corresponds to minimizing the energy function:

$$X^{\text{opt}} = \underset{X\in\mathcal{X}}{\arg\max}\,\mathbb{P}(X) = \underset{X\in\mathcal{X}}{\arg\min}\,E(X) \tag{4}$$

**Sum-Product Belief Propagation**. The key subroutine of our Spanning Tree message passing algorithms is sum-product belief propagation of Pearl (Pearl, 1988). BP is an algorithm for approximate minimization of energy $E(x)$ as in equation 3; it is exact if the graph does not have loops (*e.g.*, trees and chains). Sum-product BP maintains a directional message $M_{ij}$ from node $i$ to node $j$. The basic operation of BP is to pass the message from node $i$ to node $j$ along the edge $(i,j)$. After receiving all the messages from node $i$'s neighbors, the marginal distribution of node $i$ reads:

$$\mathbb{P}(x_i|X\backslash\{x_i\}) = \mathbb{P}(x_i)\prod_{(i,j)\in\mathcal{E}} M_{ji} \tag{5}$$

where $\mathbb{P}(x_i)$ is the prior marginal of $x_i$. In the absence of specific prior information, we will assume that all priors are uniformly distributed. BP algorithm iteratively passes messages in a specific order until a stopping criterion is met. One numerical method that improves the performance of BP is to stabilize the fixed point iteration scheme with damping (Murphy et al., 2013), which helps prevent oscillations between two steady states. This method involves replacing the term $\prod_{(i,j)} M_{ij}$ in equation 5 with a convex combination of the received messages:

$$M_{ij}^t = (1-\alpha)M_{ij}^t + \alpha M_{ij}^{t-1} \tag{6}$$

where $\alpha \in (0,1)$ is known as the damping factor and $t$ refers to the iteration number.

**Random Spanning Trees**. Given a graph $\mathcal{G} = (\mathcal{V},\mathcal{E})$, we denote $\mathcal{T}$ the collection of all the spanning trees $T$ in $\mathcal{G}$ and $\Omega(\mathcal{T})$ some distribution over $\mathcal{T}$. The spanning trees we select from $\mathcal{T}$ is $\{T_k \in \mathcal{T}|k \in \mathcal{K}\}$, where $\mathcal{K}$ is the index set of the trees. Denote $\rho_T$ the probability of sampling a spanning tree $T$ from $\mathcal{T}$. Throughout the paper, we assume that each distinct tree is sampled with equal probability, *i.e.*, for any two distinct spanning trees $T_1, T_2 \sim \Omega(\mathcal{T})$, $\rho_{T_1} = \rho_{T_2}$. For each edge $(i,j) \in \mathcal{E}$, we denote $\rho_{ij} = \mathbb{P}((i,j) \in T,\ T \sim \Omega(\mathcal{T}))$ the probability that an edge $(i,j)$ appears in a random spanning tree.

# 4 MAP INFERENCE USING TREE SAMPLING

In this section, we present our approach and examine the potential benefits of approximating the original problem by combining the results via the analysis of random spanning trees in the original graph. Furthermore, we present a heuristic that has the potential to improve the approximation outcome by capitalizing on the information obtained during the "BP – update – inference" procedure. Subsequently, we dive into a thorough examination of the underlying logic behind our algorithm and conduct a comprehensive analysis of its complexity.

**Algorithm 1:** MAP on Graph using Spanning Tree Sampling

```
Input: Graph $\mathcal{G}$, index set of spanning trees $\mathcal{K}$, number of iterations $iter$.
Output: overall states $X$.
$\mathcal{T}^{\text{cand}} \leftarrow \{T_k\}_{k=1}^{|\mathcal{K}|}$, for $T_k \sim \mathcal{T}$ ;                // sample a candidate tree set
while not converge do
    for $k \in \mathcal{K}$ do
        Apply BP to $T_k \in \mathcal{T}^{\text{cand}}$ to compute $p^t_{T_k}(x_i|X\backslash\{x_i\}) \; \forall x_i \in X$ ;
    $\mathcal{M} = \{p(x_i|X\backslash\{x_i\})|x_i \in X\} = \{\prod_{k\in\mathcal{K}} (p^t_{T_k}(x_i|X\backslash\{x_i\})|x_i \in X\}$ ;   // update marginals
    $X^{\text{Gibbs}} = \text{GibbsSampler}(\mathcal{M})$;
    $X^{\text{Greedy}} = \text{GreedySelector}(\mathcal{M})$;
    $E^{best,t}_{temp} = \min(E^{best,t-1}_{temp}, E(X^{\text{Gibbs}}), E(X^{\text{Greedy}}))$;
    Update $X^{best,t}_{temp}$ according to $E^{best,t}_{temp}$ ;                       // update best X
$X = X^{best,t}_{temp}$;
```

## 4.1 ALGORITHM

Procedures in the proposed approach are summarized in Algorithm 1 and a pipeline diagram is illustrated in Fig. 3 in **Appendix A**. While the presence of loops in the original graph renders it analytically intractable, conducting belief propagation on each of the spanning trees efficiently delivers exact solutions. Note a spanning tree is a structure that can encompass all nodes and capture a significant amount of the relationships between nodes without loops, which maximally utilizes original information and avoids the oscillations.

By sampling spanning trees from the graph and doing belief propagation on these trees, we are actually trying to formulate a series of sub-problems of the original problem and solve the original problem by solving these sub-problems. The sub-problem on a spanning tree $T$ is shown below:

$$\min_X E^T(X) = \min_{\{x_i\},\forall i\in\mathcal{V}} \left\{ \sum_{i\in\mathcal{V}} \theta_i(x_i) + \sum_{(i,j)\in\mathcal{T}} \theta_{ij}(x_i, x_j) \right\}. \tag{7}$$

To ensure that the combined energy of the spanning trees aligns with the energy of the original graph, and consequently that the joint distribution over the trees matches the joint distribution of the original graph, it is crucial to validate the correctness of the beliefs propagated within the trees. During the belief propagation procedure, messages passed along an edge represent the marginalized joint distribution of that edge, which inherently depends on the pairwise energy term.

To mitigate the bias introduced by the sampling procedure, we adjust the pairwise beliefs on each uniformly sampled spanning tree using the probability $\rho_{ij}$ of an edge $(i, j)$ appearing in a uniformly sampled spanning tree. The adjustment coefficient for the pairwise energy on edge $(i, j)$ is given by $w_{ij} = 1/\rho_{ij}$.

To accurately estimate the probability $\rho_{ij}$ of an edge appearing in a uniformly sampled spanning tree, we utilize the concept of *effective resistance*, as detailed in Section 4.3. The adjusted sub-problem on a spanning tree $T$ is formulated as follows:

$$\min_X E^T(X) = \min_{\{x_i\},\forall i\in\mathcal{V}} \left\{ \sum_{i\in\mathcal{V}} \theta_i(x_i) + \sum_{(i,j)\in\mathcal{T}} w_{ij}\theta_{ij}(x_i, x_j) \right\}. \tag{8}$$

Then we could use the energy of each sampled tree to approximate the original energy of the graph:

$$\tilde{E}(X) = \sum_{k\in\mathcal{K}} \rho_{T_k} \left[ \sum_{i\in\mathcal{V}} \theta_i(x_i) + \sum_{(i,j)\in T_k} w_{ij}\theta_{ij}(x_i, x_j) \right] = \sum_{i\in\mathcal{V}} \theta_i(x_i) + \mathbb{E}_{T\sim\Omega(\mathcal{T})}[\tilde{\Theta}(X, X)], \tag{9}$$

where $\rho_{T_k}$ is the probability that $T_k$ is sampled and $\tilde{\Theta}(X, X)$ is the pairwise energy term after reweighted. And the second term of the right-hand side is the expectation of the adjustment pairwise term according to the distribution of the random uniform spanning trees. By Lemma 1, we could make sure the merged MAP problem is actually an approximation of the original problem. We use a toy example to show the necessity of weight adjustment in **Appendix J**.

After adjustment, the incoming message of vertex $i$ from vertex $j$ now becomes $\tilde{M}_{ji} = (M_{ji})^{w_{ij}}$.

Then equation 3 on a certain tree $T$ now becomes

$$\mathbb{P}^T(X) = \frac{1}{Z^T}\exp(-E^T(X)) = \frac{1}{Z^T}\exp\left(-\sum_{i\in\mathcal{V}}\theta_i(x_i) - \sum_{(i,j)\in T} w_{ij}\theta_{ij}(x_i, x_j)\right). \tag{10}$$

Now we could approximate the joint distribution via:

$$\tilde{\mathbb{P}}(X) = \prod_{k\in\mathcal{K}} \mathbb{P}_{T_k}(X)^{\rho_{T_k}} = \frac{1}{\tilde{Z}}\exp\left(-\sum_{k\in\mathcal{K}}\rho_{T_k}\left[\sum_{i\in\mathcal{V}}\theta_i(x_i) + \sum_{(i,j)\in T_k} w_{ij}\theta_{ij}(x_i, x_j)\right]\right). \tag{11}$$

After applying belief propagation on sampled trees, we could obtain the marginals $p_{T_k}(x_i|X\backslash\{x_i\})$ on each tree. They are subsequently merged to approximate the true marginal distribution. For the initial marginals, without any specification, we will assume they are all uniform. To obtain the estimations, we do the following procedure to merge the marginals of each variables on each trees since the summation in energy is the multiplication when calculating probability:

$$\tilde{p}(x_i|X\backslash\{x_i\}) \propto \prod_{k\in\mathcal{K}}(p_{T_k^t}(x_i|X\backslash\{x_i\})). \tag{12}$$

We employed two methods, Gibbs sampling (Geman & Geman, 1984), and Greedy selection, to assign values to each of the variables based on the estimation of the marginal distributions as in Line 6 and 7 in Algorithm 1. These methods are implemented in modules named GibbsSampler and GreedySelector. The GibbsSampler samples a label for each variable based on the estimated marginal distribution, this label configuration is called $X^{Gibbs}$, while the GreedySelector selects the label with the highest value from the estimated marginal distribution, this label configuration is called $X^{Greedy}$. Since the sampling procedure will introduce uncertainty and greedy selection on nodes doesn't mean the combination is the best, we will keep recording the best configuration so far $X_{temp}^{best,t}$ during the iterations. We will update this configuration if some of the configuration given by the two modules yields the lowest energy so far $E_{temp}^{best,t}$. When the algorithm terminate, the best configuration that is found would be the final output.

**Complexity Analysis** can be found in **Appendix E**.

**Lemma 1.** *Given a uniform spanning tree distribution $\Omega(\mathcal{T})$ and the corresponding edge appearance probabilities $\{\rho_{ij}|\forall(i,j)\in\mathcal{E}\}$, when $|\mathcal{K}| = |\mathcal{T}|$, the approximation energy Eq. 9 and the original energy coincide.*

Since acquiring the true tree selection probability $\rho_T$ is intractable in practice, we use Monte Carlo Approximation to do the approximation when implementation the algorithm, by Lemma 2, you can see the condition in Lemma 1 still holds.

**Lemma 2.** *By applying the Monte Carlo Approximation, the expectation of the approximation is the original energy.*

$$\mathbb{E}_{T_k\sim\Omega(\mathcal{T})}\left[\sum_{i\in\mathcal{X}}\theta_i(x_i) + \frac{1}{|\mathcal{K}|}\sum_{k\in\mathcal{K}}\sum_{(i,j)\in T_k} w_{ij}\theta_{i,j}(x_i, x_j)\right] = \sum_{i\in\mathcal{X}}\theta_i(x_i) + \sum_{(i,j)\in\mathcal{E}}\theta_{ij}(x_i, x_j) \tag{13}$$

**Theorem 1.** *Given spanning tree distribution $\Omega(\mathcal{T})$ and the corresponding edge appearance probability $\{\rho_{ij}|\forall(i,j)\in\mathcal{E}\}$, the following error bound of the approximation energy Eq. equation 9 holds with probability at least $1-\delta$.*

$$|E(X) - \tilde{E}(X)| \leq \sqrt{\frac{1}{|\mathcal{K}|}\sum_{(i,j)\in\mathcal{E}}\theta_{ij}^2(x_i, x_j)(\frac{1-\rho_{ij}}{\rho_{ij}})}\frac{1}{\sqrt{\delta}}. \tag{14}$$

*When*

$$|\mathcal{K}| \geq \frac{1}{\delta\eta^2}\sum_{(i,j)\in\mathcal{E}}\theta_{ij}^2(x_i, x_j)(\frac{1-\rho_{ij}}{\rho_{ij}}), \tag{15}$$

*we have $P(|E(X) - \tilde{E}(X)| \geq \eta) \leq \delta$.*

The proof of Lemma 1, Lemma C and Theorem 1 can be found in Appendix B and Appendix D respectively. It is obvious that due to the term $(\frac{1-\rho_{ij}}{\rho_{ij}})$, we can achieve good quality results with only a few trees when the graph is sparse.

The error bound exhibits an inverse relationship with the edge selection probability $\rho_{ij}$. As $\rho_{ij}$ approaches smaller values, the factor $(\frac{1-\rho_{ij}}{\rho_{ij}})$ grows significantly, leading to a looser error bound. This relationship provides insight into the performance disparity between sparse and dense graphs. In sparse graphs, each edge typically has a higher probability of being included in a spanning tree, since fewer alternative paths exist between vertices. Conversely, in dense graphs, the abundance of potential paths results in lower individual edge selection probabilities. Consequently, when $\rho_{ij}$ is larger (as in sparse graphs), the term $(\frac{1-\rho_{ij}}{\rho_{ij}})$ remains relatively small, yielding a tighter error bound and better estimation accuracy. Moreover, this theoretical analysis aligns with our empirical observations of superior performance on sparse graphs. This mathematical relationship explains why our method naturally performs better on sparse structures, where the higher edge selection probabilities contribute to more reliable estimates.

### 4.2 Random Uniform Spanning Tree

The key idea behind our algorithm is utilizing the partial information contained inside the original graphs. To uniformly sample a random spanning tree from the given graph, we employed the widely used Wilson's algorithm (Wilson, 1996). This algorithm, based on random walks and removal, ensures the uniformity of the resulting spanning tree selection. In general, the time complexity of Wilson's algorithm is $O(N^3)$. It could be significant when the size of the graph increases. Using methods such as Depth-First Search (DFS) would be more efficient, given the time complexity of $O(M+N)$. However, these methods make it challenging to identify the distribution of the trees. As a result, we cannot adjust the messages based on the edge selection probabilities. Therefore, one of the future trials is to incorporate advanced (approximate) sampling procedures such as Schild (2018) to reduce the time complexity.

### 4.3 Effective Resistance

We employ algorithms approximating effective resistance to obtain $\rho_{ij}$. Effective resistance is used in electrical network analysis and graph theory to measure the resistance between two vertices in a graph (Lyons & Peres, 2016) – how difficult it is for current to flow between two points in a network. The terminology of effective resistance originates from the following observation: Given the resistance on all edges, if one removes all vertices of $\mathcal{G}$ except $(s,t)$ and replaces the whole network with a resistance of resistor $\mathrm{Reff}(s,t)$ between $(s,t)$, then, the energy (and the potential difference) of all electrical flows between $(s,t)$ remains invariant.

There is a strong connection between $\rho_{ij}$ and the corresponding effective resistance. By applying Kirchhoff's effective resistance formula (Lyons & Peres, 2016), we can establish the following lemma.

**Lemma 3** ( (Madry et al., 2014; Doyle & Snell, 2000)). *For any unweighted graph* $\mathcal{G}=(\mathcal{V},\mathcal{E})$*, any edge* $(i,j)\in\mathcal{E}$*,* $\rho_{ij} = \mathbb{P}[(i,j)\in T, T\sim\Omega(\mathcal{T})] = \mathrm{Reff}(i,j)$*.*

As such, we resort to an efficient alternative algorithm approximating effective resistance (Vos, 2016) for $\rho_{ij}$. In our research, we only need the appearance probabilities of all the edges to calculate the weight of the pairwise energies, so we only need to calculate the effective resistance between directly connected nodes instead of do the calculation between all the nodes. **Note**, the value $\mathrm{Reff}(i,j)$ represents the probability of *sampling* edge $(i,j)$ when reaching node $i$ or node $j$ in Wilson's algorithm. In this sense, it is unnecessary to directly reweight the pairwise terms $\theta_{ij}$. More details are in **Appendix F**.

## 5 Experiments

We present a comprehensive investigation into the performance of the variational inference method Mean-field and three belief-based inference algorithms: the Loopy Belief Propagation algorithm (LBP), the Tree-reweighted Belief Propagation algorithm (TRBP), and a novel sampling algorithm proposed in this study, referred to as **SPT**. The experiments aim to validate the arguments presented in the previous sections and establish the superiority of our proposed algorithm over LBP and TRBP. Performance evaluations are conducted on synthetic problems with diverse graph structures, encompassing energy functions such as the square function $\theta_{ij}=\alpha(x_i-x_j)^2$, absolute difference function $\theta_{ij}=\alpha|x_i-x_j|$ and Potts function $\theta_{ij}=\alpha(x_i=x_j)+\beta$, where $\alpha$ and $\beta$ are constant penalty terms. In this section, we use $\alpha=1$ and $\beta=0.1$.

Experiments involve grid-based and cell-based structures, as well as Erdős-Rényi (ER) random graphs. The LBP algorithm and TRBP algorithm are from Kappes et al. (2015); Andres et al. (2012); Maxwell Forbes (2017). TRBP here is the refined version of TRW-based algorithms. The mean-field algorithm we used is from

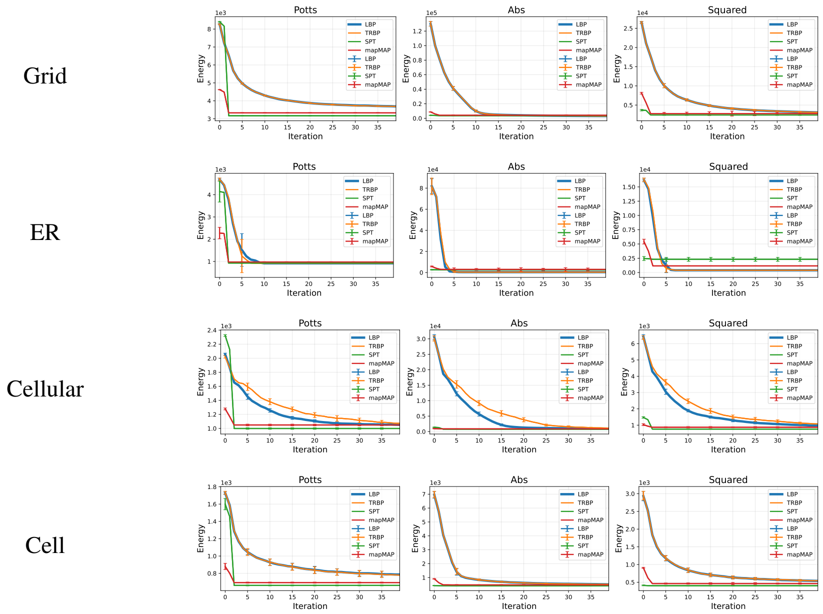


Figure 1: Results on Grid, Erdős-Rényi (ER), Cellular, and Cell graph types (rows) with Potts, Absolute difference (Abs), and Squared difference energies. Each plot shows energy $E(x|\bar{\theta})$ vs. iterations. Blue: LBP, orange: TRBP, green: SPT. The error bars show the standard deviation of each algorithm. Note: LBP and TRBP curves sometimes overlap due to minimal differences.

Table 1: Results on real PCI instances. Numbers are obtained energy values. Best in bold.

| GRAPH | #VAR/#CON | #NODES/#EDGES | LBP | TRBP | MAPMAP | SPT |
|---|---|---|---|---|---|---|
| PCI_INSTANCE_1 | 955/2496 | 30/165 | 3.72662E+08 | 3.72662E+08 | 101259.1 | **84382.6** |
| PCI_INSTANCE_2 | 1588/4409 | 40/311 | 3.72704E+08 | 3.72704E+08 | 214875 | **186848** |
| PCI_INSTANCE_3 | 17684/52673 | 80/1522 | 0.303468 | 0.303468 | 0.299465 | **0.295245** |
| PCI_INSTANCE_4 | 65713/193287 | 286/10565 | 0.751074 | 0.751074 | 0.725626 | **0.552074** |

Wang et al. (2022) and the mapMAP solver is from Thuerck et al. (2016). Furthermore, the applicability of our algorithm on real-world MRF inference datasets is demonstrated.

## 5.1 SYNTHETIC PROBLEMS

We first conduct experiments on synthetic problem instances. To test the applicability of the proposed method in a wide spectrum of settings, we test four types of graph topologies: Grid, Erdős-Rényi (ER), Cellular, and Cell. There are illustrative figures in **Appendix P.1** visualizing typical layouts of cell graphs (Fig. 7(a)) and cellular graphs (Fig. 7(b)). All the experimental results are in Fig. 1. Concrete settings are as follows.

**Grid.** We first test these algorithms on 70×70 grids with 4900 nodes, and each random node has 5 possible discrete labels to select. In addition, each label is attached to an explicit energy generated randomly and the energy $e_l \in (0, 1)$. For the setting of LBP and TRBP, we set the maximum round of iterations to 40 rounds. The damping factor of both of them is 0.8 since it is the best configuration as we could see from the results of the experiments. For our SPT, the maximum round of iterations is set to 20 and the damping factor is 0. We run the experiment for 10 different test cases and the final result is the average result over these cases.

**Erdős-Rényi.** Random MRF instances under Erdős-Rényi setting comprise 2,500 nodes, with an average degree of 12. The test cases are configured with different energy functions, and all the unary energy terms are generated randomly. In order to show the inference ability clearly, we attach 10 possible labels for each node in this part. The damping factors are set to 0 in this part of experiment. The maximum iterations, including our SPT algorithm are all set to 40. The number of trees used by SPT for the tests is 20.

**Cellular & Cell.** In addition to these two most common type of networks. We then run the tests on cellular and cell networks which are widely applied in communication networks. Each of the cellular networks has 4,998 nodes and 7,398 edges, the average degree is about 2.96. Each of the cell networks has 4,900 nodes and 14,421 edges, the average degree is about 5.89. The other settings are as same as we used for the experiments on Erdős-Rényi graphs.

Table 2: Results on the UAI dataset. Numbers are the obtained energy values. Best in bold.

| GRAPH | #NODES/#EDGES | Mean-field | | LBP | | TRBP | | mapMAP | | SPT | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| | | Energy | Time(s) | Energy | Time(s) | Energy | Time(s) | Energy | Time(s) | Energy | Time(s) |
| ProteinFolding_11 | 400/7160 | 1.0414e+08 | 0.0343 | **7.11675e+07** | 0.880201 | 7.54224e+07 | 0.967124 | 8.54620e+07 | 0.04 | 7.49639e+07 | 13.2087 ($\rho_{ij}$ time 1.66887) |
| ProteinFolding_12 | 250/1848 | 5852.2148 | 2.2345 | **0.949** | 13.4593 | **0.949** | 15.4034 | **0.949** | 0.106 | **0.949** | 15.9054 ($\rho_{ij}$ time 0.448073) |
| Grids_19 | 1600/3200 | 10613.0645 | 0.1503 | 7053.72 | 3.32058 | **6056.45** | 3.66748 | 7523.18 | 0.093 | 6275.56 | 112.945 ($\rho_{ij}$ time 89.1308) |
| Grids_21 | 1600/3200 | 3.3756e+08 | 0.1598 | 2.5577e+08 | 3.30869 | **2.56852e+08** | 3.6697 | 3.33740e+08 | 0.141 | 3.22289e+08 | 137.674($\rho_{ij}$ time 89.106) |
| Grids_24 | 1600/3120 | 3.8874e+08 | 0.1423 | 3.08505e+08 | 3.22656 | **3.01668e+08** | 3.56594 | 3.53902e+08 | 0.046 | 3.09172e+08 | 142.921 ($\rho_{ij}$ time 89.0508) |
| Grids_25 | 1600/3120 | 10487.6113 | 0.1337 | 5476.03 | 3.099 | **5441.52** | 3.47539 | 7635.35 | 0.071 | 5931.11 | 115.198 ($\rho_{ij}$ time 90.8171) |
| Grids_26 | 400/800 | 818490.187 | 0.0317 | 792987 | 0.824318 | **556629** | 0.914945 | 794293 | 0.048 | 759351.5 | 12.3774($\rho_{ij}$ time 1.61949s) |
| Grids_27 | 1600/3120 | 3.892e+6 | 0.1375 | **2.06277e+06** | 3.22343 | 2.14945e+06 | 3.57305 | 3.54505e+06 | 0.105 | 3.41471e+06 | 132.454 ($\rho_{ij}$ time 89.1219) |
| Grids_30 | 400/760 | 1044407.0625 | 0.0313 | **567339** | 0.7504 | 663905 | 0.831959 | 817746 | 0.04 | 809344 | 13.4867 ($\rho_{ij}$ time 1.65657) |
| Segmentation_11 | 228/624 | 401.231 | 0.2086 | 346.647 | 24.8519 | 348.097 | 29.679 | 286.022 | 0.024 | **200.866** | 12.2192 ($\rho_{ij}$ time 0.366381) |
| Segmentation_12 | 231/625 | 805.6727 | 0.0212 | 735.259 | 0.822135 | 735.259 | 0.899114 | 724.025 | 0.018 | **611.347** | 3.72848 ($\rho_{ij}$ time 0.377893) |
| Segmentation_13 | 225/607 | 785.7468 | 0.0208 | 726.144 | 0.632035 | 726.144 | 0.708431 | 7689.666 | 0.027 | **596.606** | 3.05409 ($\rho_{ij}$ time 0.347453) |
| Segmentation_14 | 231/632 | 803.2781 | 0.229 | 742.241 | 0.752022 | 742.241 | 0.824835 | 684.055 | 0.021 | **629.95** | 3.63654 ($\rho_{ij}$ time 0.375578) |
| Segmentation_15 | 229/622 | 314.47 | 0.1961 | 362.61 | 24.764 | 362.61 | 29.4172 | 277.087 | 0.023 | **191.924** | 9.8776 ($\rho_{ij}$ time 0.364987) |
| Segmentation_16 | 228/610 | 776.0317 | 0.211 | 720.009 | 0.811115 | 720.009 | 0.886003 | 647.788 | 0.026 | **578.114** | 3.53684 ($\rho_{ij}$ time 0.363007) |
| Segmentation_17 | 225/612 | 516.1681 | 0.1895 | 392.83 | 25.8389 | 370.852 | 30.3104 | 296.107 | 0.022 | **187.432** | 9.67083 ($\rho_{ij}$ time 0.34816) |
| Segmentation_18 | 235/647 | 820.7836 | 0.0235 | 782.282 | 0.745899 | 782.282 | 0.825011 | 662.122 | 0.031 | **595.11** | 3.68485 ($\rho_{ij}$ time 0.396227) |
| Segmentation_19 | 228/624 | 798.2877 | 0.221 | 742.197 | 0.72549 | 742.197 | 0.807067 | 660.399 | 0.023 | **624.649** | 3.39231($\rho_{ij}$ time 0.360591) |
| Segmentation_20 | 232/635 | 578.8339 | 0.3630 | 373.14 | 26.832 | 371.839 | 31.4758 | 287.187 | 0.023 | **196.558** | 10.7515 ($\rho_{ij}$ time 0.368231) |

**Results & Analysis.** The results are displayed in Fig. 1. These charts show our algorithm effectively handles various energy functions on graphs with specific local structures and randomly generated graphs. For grid, cellular, and cell graphs (all locally and sparsely connected), our algorithm outperforms both LBP, TRBP and mapMAP in final energy and iteration efficiency. These structures commonly appear in wireless communication and traffic control industries, where strategically placed interconnected devices create potential interference patterns similar to our experimental setup. Our algorithm performs consistently well on completely randomly generated graphs, even when graph density is approximately twice that of cellular networks. In addition, the error bars indicate that these algorithms remain highly stable after convergence, with our SPT algorithm exhibiting particularly robust performance. However, it slightly underperforms compared to LBP and TRBP when using absolute difference as the energy function. This occurs because when different choices have similar costs, the algorithm may converge to sub-optimal positions.

### 5.2 UAI INFERENCE COMPETITION

We extend evaluation to factor graphs from the 2022 UAI Inference Competition's MAP tasks[1]. Table 2 compares Mean-field, LBP, TRBP, and our SPT method, reporting final energy values. Algorithm settings match those used on synthetic problems, except SPT uses 20 trees for estimation in this section. Overall, SPT achieves competitive performance comparable to LBP and TRBP, with significant improvements on various segmentation tasks. On segmentation instances, SPT outperforms all other methods. These instances have graph structures similar to those in the synthetic experiments and share the same Potts model characteristics. For ProteinFolding_11 and Grid cases, our algorithm's performance matches LBP and TRBP, consistent with findings from Grid graph experiments in the synthetic problem set. Graph structures for ProteinFolding_11, ProteinFolding_12, Segmentation_14 and Grids_30 instances are shown in Appendix P.2. We display only one Segmentation class instance since they all share similar graph structure.

### 5.3 REAL-WORLD PCI PROBLEM OF 5G NETWORKS

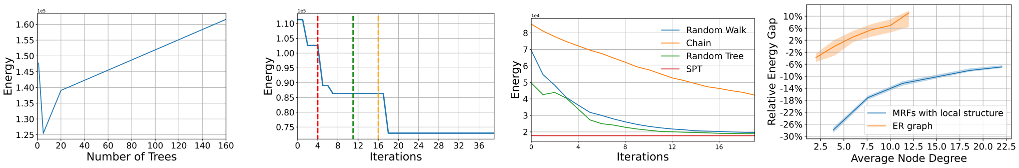
PCI (Physical Cell Identity) uniquely identifies cells in LTE and 5G networks. We evaluate using internal PCI data transformed into pairwise MRFs for MAP inference. Appendix L details the MIP formulation and transformation process. We evaluate our algorithm on four internal *real industry-level PCI instances* compared to LBP and TRBP, using final energy (corresponding to the objective function in equation 52) as our comparison standard. Results appear in Table 1 with instance topologies in Appendix P.3. When MAP estimation is formulated as integer programming, problem size increases significantly, as shown in the table. Both LBP and SPT yield identical results, but our algorithm achieves superior performance compare to all the baselines, with substantial energy reductions for PCI_INSTANCE_1 and PCI_INSTANCE_2. The inference time are shown in Table. 4 in Appendix. K.

### 5.4 MORE ANALYSIS

To investigate the impact of various factors, we examine the following aspects: the number of trees used, the number of iterations, stopping criteria, other sub-graph structures, and impact of sparsity. "The main part of the result figures in this section is displayed in Fig. 2.

**Number of trees**. We evaluated our algorithm's performance on the PCI_INSTANCE_2 model using 1, 5, 20, 80, 160 spanning trees. After 10 iterations per configuration, the final energy values revealed a V-shaped curve, with optimal performance at 5 spanning trees (Fig. 2a). This suggests that for complicate graphs, approximately 5-10 spanning trees provide sufficient structure representation without redundant information. As

[1] https://www.auai.org/uai2022/uai2022_competition



(a) Energy v.s. tree numbers (b) Energy v.s. iterations (c) Performance by different subgraphs (d) SPT-LBP gap vs. sparsity

Figure 2: Part of the results for the analysis in Section. 5.4.

demonstrated in Appendix I, for trivial MRFs with known optimal energies, increasing the number of trees can further minimize the optimality gap.

**Number of iterations & Stopping criterion**. Fig. 2b shows energy curves for our algorithm on the PCI_INSTANCE_1 dataset using 20 spanning trees over 40 iterations. The optimal configuration is achieved at iteration 19 and persists until the end. Notably, the provisional best configuration remains stable during several periods within the first 20 iterations. Common early stopping criteria based on monitoring consecutive iterations without improvement (marked by red, green, and yellow dotted lines at thresholds of 3, 5, and 10 iterations) would fail to capture the best solution found at iteration 19. Given our algorithm's efficiency, we recommend running it for the maximum number of iterations in practice.

**Inference Time**. In Table 2, we show both overall inference time and edge selection probability calculation time (marked as "$\rho_{ij}$ time 1.324"). For complex problems like "Segmentation_11", all methods require more computation time, but SPT demonstrates more stable performance and faster convergence. Our analysis identifies calculating edge selection probabilities as the main computational bottleneck, especially for larger graphs. Developing more efficient probability estimation methods is an important future research direction. The algorithm could be accelerated through parallelization, with details available in Appendix N

**Other sub-graph structures**.We evaluated several structural alternatives: chains, random trees (covering 75% of nodes), and random walks (1600 steps for greater node coverage). Testing details are in Appendix. H. Tests used 1600-node grids with 10-16 possible states per node and square function pairwise energy without unary terms. Unlike our fixed-tree SPT approach, alternative structures were resampled each iteration since they don't guarantee full node coverage. Running 20 iterations with SPT using 10 trees, Fig. 2c shows SPT achieved superior results with significantly fewer iterations. Random trees and walks delivered acceptable results but required more iterations to converge. Comparison with the tree-coupling approach from Hamze & de Freitas (2004) appears in Appendix G.

**Impact of sparsity**. From Fig.1, our algorithm outperforms LBP and TRBP on both grid and cell structures, with a larger performance gap on denser cell graphs. To evaluate effectiveness on locally and sparsely connected MRFs (Fig.8, Appendix P) and analyze local structure complexity impact, we tested LBP and SPT on graphs described in Appendix M. These graphs had average degrees of 3.8, 7.6, 11.3, 18.5, and 22.0. SPT used 10 trees, with 10-16 states per random variable and nodes linked to random observations. Pairwise energy used the Potts function. Performance comparison used the relative energy gap (equation 16), averaged over 10 trials.

$$\text{GAP} = (E_{SPT} - E_{LBP})/E_{LBP} \times 100\% \tag{16}$$

Fig. 2d shows that as local structures become denser, SPT increasingly outperforms LBP. This occurs because loop impacts become more significant, making it crucial to balance variable values within local graph structures—a strength of our algorithm. The shaded areas indicate modest gap variances.

Our algorithm also performs effectively on random graphs, especially when they're not highly dense. We conducted similar experiments on ER graphs with 900 nodes, varying average node degrees from 2 to 12 using squared energy functions (as performance with Potts functions is similar). Results in Fig. 2d (orange line) show our algorithm performs effectively on sparse graphs, even without specific graph topologies. The algorithm achieves satisfactory results on sparse graphs, but as average node degree increases, the performance gap between our algorithm and LBP increases linearly. Without local structure to leverage, SPT's performance is reduced.

## 6 Conclusion

MAP inference on MRFs remains a persistent challenge. We propose a novel, efficient approach for MAP estimation on locally and sparsely connected MRFs using spanning tree sampling. Our method works by solving MAP problems on individual sampled trees and merging these solutions into a final configuration. Experiments demonstrate superior performance against strong baselines across diverse settings, with particularly significant improvements on real-world PCI instances, indicating broad applicability potential.

## 7 REPRODUCIBILITY STATEMENT

To ensure the reproducibility of our research findings, we have taken the following comprehensive measures:

Our implementation code has been made publicly available, with the repository link provided at the end of the abstract. This allows other researchers to directly access and utilize our algorithm for their own applications or verification purposes.

Detailed instructions for setting up and running our code are included in the repository documentation. These instructions cover all necessary steps from environment setup to execution of experiments, enabling smooth replication of our work.

In Section 5, we have explicitly documented all algorithm settings used in our experiments, including parameter configurations, optimization choices, and implementation details. These specifications are critical for reproducing our experimental results.

Access information for the UAI Inference Competition datasets used in our evaluation is clearly provided in Section 5.2. This includes source links and preprocessing steps applied to the data before running our experiments.

For synthetic experiments, we have thoroughly documented our instance generation methodology in Section 5.4, with additional technical details available in Appendix H. This documentation ensures that synthetic test cases can be precisely recreated.

We provide comprehensive instructions for transforming PCI problems from their original Mixed Integer Programming (MIP) format into the MRF format required by our algorithm. This transformation process is crucial for applying our method to real-world telecommunication network optimization problems.

Through these measures, we have ensured that all aspects of our research can be independently verified and built upon by the research community.

## 8 ETHICS STATEMENT

During the preparation and submission of this paper, we have strictly adhered to the Code of Ethics in scientific research. We ensured proper citation of all relevant work, maintained integrity in our experimental procedures, reported results accurately without manipulation, and respected confidentiality of data sources where applicable. All authors have contributed substantially to this work and approved the final manuscript, with no conflicts of interest undisclosed.

# REFERENCES

Uai 2022 competition. https://uaicompetition.github.io/uci-2022/.

Mamta Agiwal, Abhishek Roy, and Navrati Saxena. Next generation 5g wireless networks: A comprehensive survey. *IEEE Communications Surveys  Tutorials*, 18(3):1617–1655, 2016. doi: 10.1109/COMST.2016.2532458.

S.M. Aji and R.J. McEliece. The generalized distributive law. *IEEE Transactions on Information Theory*, 46 (2):325–343, 2000. doi: 10.1109/18.825794.

B. Andres, T. Beier, and J.H. Kappes. OpenGM: A C++ library for discrete graphical models. *CoRR*, abs/1206.0111, 2012. URL http://arxiv.org/abs/1206.0111.

MOSEK ApS. *The MOSEK Python Fusion API manual. Version 11.0.*, 2025. URL https://docs.mosek.com/latest/pythonfusion/index.html.

Dhruv Batra, A. C. Gallagher, Devi Parikh, and Tsuhan Chen. Beyond trees: Mrf inference via outer-planar decomposition. In *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 2496–2503, 2010. doi: 10.1109/CVPR.2010.5539951.

Julian Besag. Spatial interaction and the statistical analysis of lattice systems. *Journal of the Royal Statistical Society. Series B (Methodological)*, 36(2):192–236, 1974. ISSN 00359246. URL http://www.jstor.org/stable/2984812.

Y. Boykov, O. Veksler, and R. Zabih. Markov random fields with efficient approximations. In *Proceedings. 1998 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No.98CB36231)*, pp. 648–655, 1998. doi: 10.1109/CVPR.1998.698673.

Joseph K. Bradley and Carlos Guestrin. Learning tree conditional random fields. In *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, pp. 127–134, Madison, WI, USA, 2010. Omnipress. ISBN 9781605589077.

Nicolò Cesa-Bianchi, Claudio Gentile, Fabio Vitale, and Giovanni Zappella. Random spanning trees and the prediction of weighted graphs. pp. 175–182, 01 2010.

P Clifford and JM Hammersley. Markov fields on finite graphs and lattices, 1971.

Paul Cuffe and Andrew Keane. Visualizing the electrical structure of power systems. *IEEE Systems Journal*, 11(3):1810–1821, 2017.

Peter G. Doyle and J. Laurie Snell. Random walks and electric networks, 2000. URL https://arxiv.org/abs/math/0001057.

Pedro F. Felzenszwalb and Daniel P. Huttenlocher. Efficient belief propagation for early vision. *International Journal of Computer Vision*, 70:41–54, 2004. URL https://api.semanticscholar.org/CorpusID:8702465.

Brendan Frey and David Mackay. A revolution: Belief propagation in graphs with cycles. 08 2002.

Stuart Geman and Donald Geman. Stochastic relaxation, gibbs distributions, and the bayesian restoration of images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 6(6):721–741, 1984.

D.M. Greig, B.T. Porteous, and Allan Seheult. Exact maximum a posteriori estimation for binary images. *Journal of the Royal Statistical Society, Series B*, 51:271–279, 01 1989. doi: 10.1111/j.2517-6161.1989.tb01764.x.

Anna Grim and Pedro Felzenszwalb. Convex combination belief propagation. *Applied Mathematics and Computation*, 438:127572, 2023. ISSN 0096-3003. doi: https://doi.org/10.1016/j.amc.2022.127572. URL https://www.sciencedirect.com/science/article/pii/S0096300322006464.

Firas Hamze and Nando de Freitas. From fields to trees. In *Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence*, pp. 243–250, 2004.

H. Ishikawa. Exact optimization for markov random fields with convex priors. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 25(10):1333–1336, 2003. doi: 10.1109/TPAMI.2003.1233908.

H. Ishikawa and D. Geiger. Segmentation by grouping junctions. In *Proceedings. 1998 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (Cat. No.98CB36231)*, pp. 125–131, 1998. doi: 10.1109/CVPR.1998.698598.

JörgH. Kappes, Bjoern Andres, FredA. Hamprecht, Christoph Schnörr, Sebastian Nowozin, Dhruv Batra, Sungwoong Kim, BernhardX. Kausler, Thorben Kröger, Jan Lellmann, Nikos Komodakis, Bogdan Savchynskyy, and Carsten Rother. A comparative study of modern inference techniques for structured discrete energy minimization problems. *International Journal of Computer Vision*, 115(2):155–184, 2015. ISSN 0920-5691. doi: 10.1007/s11263-015-0809-x.

Alec Kirkley, George T. Cantwell, and M. E. J. Newman. Belief propagation for networks with loops. *Science Advances*, 7(17):eabf1211, 2021. doi: 10.1126/sciadv.abf1211. URL https://www.science.org/doi/abs/10.1126/sciadv.abf1211.

V. Kolmogorov. Convergent Tree-Reweighted Message Passing for Energy Minimization. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(10):1568–1583, October 2006. ISSN 0162-8828, 2160-9292. doi: 10.1109/TPAMI.2006.200. URL http://ieeexplore.ieee.org/document/1677515/.

Nikos Komodakis, Nikos Paragios, and Georgios Tziritas. Mrf optimization via dual decomposition: Message-passing revisited. pp. 1–8, 11 2007. ISBN 978-1-4244-1631-8. doi: 10.1109/ICCV.2007.4408890.

Arun Kumar, Aniruddha A. Dutta, Nayan Jain, and Sanjay K. Dhurandher. Mrfbp: Markov random field-driven spatial entropy-based routing protocol in opportunistic networks. *International Journal of Communication Systems*, 35(1), January 2022. ISSN 1074-5351. doi: 10.1002/dac.5018. Publisher Copyright: © 2021 John Wiley Sons Ltd.

Sanjiv Kumar, Jonas August, and Martial Hebert. Exploiting inference for approximate parameter learning in discriminative fields: An empirical study. In Anand Rangarajan, Baba Vemuri, and Alan L. Yuille (eds.), *Energy Minimization Methods in Computer Vision and Pattern Recognition*, pp. 153–168, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg. ISBN 978-3-540-32098-2.

Yixin Li, Chen Li, Xiaoyan Li, Kai Wang, Md Mamunur Rahaman, Changhao Sun, Hao Chen, Xinran Wu, Hong Zhang, and Qian Wang. A comprehensive review of markov random field and conditional random field approaches in pathology image analysis. *Archives of Computational Methods in Engineering*, 29:609 – 639, 2020. URL https://api.semanticscholar.org/CorpusID:221995730.

Zhiliang Liu and Zengzhi Zou. Analysis of network topology and deployment mode of 5g wireless access network. *Computer Communications*, 160:34–42, 2020. ISSN 0140-3664. doi: https://doi.org/10.1016/j.comcom.2020.05.045. URL https://www.sciencedirect.com/science/article/pii/S014036642030654X.

Russell Lyons and Yuval Peres. *Probability on Trees and Networks*, volume 42 of *Cambridge Series in Statistical and Probabilistic Mathematics*. Cambridge University Press, New York, 2016. ISBN 978-1-107-16015-6.

Aleksander Madry, Damian Straszak, and Jakub Tarnawski. Fast generation of random spanning trees and the effective resistance metric. In *Proceedings of the Twenty-Sixth Annual ACM-SIAM Symposium on Discrete Algorithms*. Society for Industrial and Applied Mathematics, December 2014. doi: 10.1137/1.9781611973730.134. URL http://dx.doi.org/10.1137/1.9781611973730.134.

Sadia Majeed, Muhammad Uzair, Usman Qamar, and Aftab Farooq. Social network analysis visualization tools: A comparative review. In *2020 IEEE 23rd International Multitopic Conference (INMIC)*, pp. 1–6, 2020.

Yan Wong. Maxwell Forbes. py-factorgraph. https://github.com/mbforbes/py-factorgraph, 2017.

Andrea Montanari and Tommaso Rizzo. How to compute loop corrections to the bethe approximation. *Journal of Statistical Mechanics: Theory and Experiment*, 2005:P10011 – P10011, 2005. URL https://api.semanticscholar.org/CorpusID:15437986.

Kevin Murphy, Yair Weiss, and Michael I. Jordan. Loopy belief propagation for approximate inference: An empirical study, 2013.

Judea Pearl. Reverend bayes on inference engines: a distributed hierarchical approach. In *Proceedings of the Second AAAI Conference on Artificial Intelligence*, AAAI'82, pp. 133–136. AAAI Press, 1982.

Judea Pearl. Chapter 4 - belief updating by network propagation. In Judea Pearl (ed.), *Probabilistic Reasoning in Intelligent Systems*, pp. 143–237. Morgan Kaufmann, San Francisco (CA), 1988. ISBN 978-0-08-051489-5. doi: https://doi.org/10.1016/B978-0-08-051489-5.50010-2. URL https://www.sciencedirect.com/science/article/pii/B9780080514895500102.

Patrick Pletscher, Cheng Soon Ong, and Joachim Buhmann. Spanning tree approximations for conditional random fields. In David van Dyk and Max Welling (eds.), *Proceedings of the Twelfth International Conference on Artificial Intelligence and Statistics*, volume 5 of *Proceedings of Machine Learning Research*, pp. 408–415, Hilton Clearwater Beach Resort, Clearwater Beach, Florida USA, 16–18 Apr 2009. PMLR. URL `https://proceedings.mlr.press/v5/pletscher09a.html`.

S. Roy and I.J. Cox. A maximum-flow formulation of the n-camera stereo correspondence problem. In *Sixth International Conference on Computer Vision (IEEE Cat. No.98CH36271)*, pp. 492–499, 1998. doi: 10.1109/ICCV.1998.710763.

Masaki Saito, Takayuki Okatani, and Koichiro Deguchi. Application of the mean field methods to mrf optimization in computer vision. In *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 1680–1687, 2012. doi: 10.1109/CVPR.2012.6247862.

Vladimir Savic and Santiago Zazo. Nonparametric belief propagation based on spanning trees for cooperative localization in wireless sensor networks. In *2010 IEEE 72nd Vehicular Technology Conference - Fall*, pp. 1–5, 2010. doi: 10.1109/VETECF.2010.5594105.

Aaron Schild. An almost-linear time algorithm for uniform random spanning tree generation. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2018, pp. 214–227, New York, NY, USA, 2018. Association for Computing Machinery. ISBN 9781450355599. doi: 10.1145/3188745.3188852. URL `https://doi.org/10.1145/3188745.3188852`.

Dmitrij Schlesinger and BORIS FLACH. Transforming an arbitrary minsum problem into a binary one. 01 2006.

Alexei N. Skurikhin. Learning tree-structured approximations for conditional random fields. In *2014 IEEE Applied Imagery Pattern Recognition Workshop (AIPR)*, pp. 1–8, 2014. doi: 10.1109/AIPR.2014.7041937.

Wenkang Su, Jiangqun Ni, Xianglei Hu, and Jessica Fridrich. Image steganography with symmetric embedding using gaussian markov random field model. *IEEE Transactions on Circuits and Systems for Video Technology*, 31(3):1001–1015, 2021. doi: 10.1109/TCSVT.2020.3001122.

Jian Sun, Heung-Yeung Shum, and Nan-Ning Zheng. Stereo matching using belief propagation. In *Proceedings of the 7th European Conference on Computer Vision-Part II*, ECCV '02, pp. 510–524, Berlin, Heidelberg, 2002. Springer-Verlag. ISBN 3540437444.

Martin Szummer, Pushmeet Kohli, and Derek Hoiem. Learning crfs using graph cuts. volume 5303, pp. 582–595, 10 2008. ISBN 978-3-540-88685-3. doi: 10.1007/978-3-540-88688-4_43.

Daniel Thuerck, Michael Waechter, Sven Widmer, Max von Buelow, Patrick Seemann, Marc E. Pfetsch, and Michael Goesele. A fast, massively parallel solver for large, irregular pairwise Markov random fields. In *Proceedings of High Performance Graphics 2016*, 2016.

Vaya Sapobi Samui Vos. *Methods for determining the effective resistance*. Master's thesis, Mathematisch Instituut Universiteit Leiden, 2016.

M.J. Wainwright, T.S. Jaakkola, and A.S. Willsky. Map estimation via agreement on trees: message-passing and linear programming. *IEEE Transactions on Information Theory*, 51(11):3697–3717, 2005. doi: 10.1109/TIT.2005.856938.

Chaohui Wang, Nikos Komodakis, and Nikos Paragios. Markov random field modeling, inference learning in computer vision image understanding: A survey. *Computer Vision and Image Understanding*, 117(11): 1610–1627, 2013. ISSN 1077-3142. doi: https://doi.org/10.1016/j.cviu.2013.07.004. URL `https://www.sciencedirect.com/science/article/pii/S1077314213001343`.

Lihao Wang, Yi Zhou, Yiqun Wang, Xiaoqing Zheng, Xuanjing Huang, and Hao Zhou. Regularized molecular conformation fields. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho (eds.), *Advances in Neural Information Processing Systems*, 2022. URL `https://openreview.net/forum?id=7XCFxnG8nGS`.

Y. Weiss and W.T. Freeman. On the optimality of solutions of the max-product belief-propagation algorithm in arbitrary graphs. *IEEE Transactions on Information Theory*, 47(2):736–744, 2001. doi: 10.1109/18.910585.

Wim Wiegerinck and Tom Heskes. Fractional belief propagation. In S. Becker, S. Thrun, and K. Obermayer (eds.), *Advances in Neural Information Processing Systems*, volume 15. MIT Press, 2002. URL `https://proceedings.neurips.cc/paper_files/paper/2002/file/35936504a37d53e03abdfbc7318d9ec7-Paper.pdf`.

David Bruce Wilson. Generating random spanning trees more quickly than the cover time. In *Symposium on the Theory of Computing*, 1996. URL https://api.semanticscholar.org/CorpusID:207198080.

Yongji Wu, Defu Lian, Yiheng Xu, Le Wu, and Enhong Chen. Graph convolutional networks with markov random field reasoning for social spammer detection. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(01):1054–1061, Apr. 2020. doi: 10.1609/aaai.v34i01.5455. URL https://ojs.aaai.org/index.php/AAAI/article/view/5455.

Bingbing Xu, Huawei Shen, Bingjie Sun, Rong An, Qi Cao, and Xueqi Cheng. Towards consumer loan fraud detection: Graph neural networks with role-constrained conditional random field. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(5):4537–4545, May 2021a. doi: 10.1609/aaai.v35i5.16582. URL https://ojs.aaai.org/index.php/AAAI/article/view/16582.

Zhiwei Xu, Thalaiyasingam Ajanthan, and Richard Hartley. Fast and differentiable message passing on pairwise markov random fields. In Hiroshi Ishikawa, Cheng-Lin Liu, Tomas Pajdla, and Jianbo Shi (eds.), *Computer Vision – ACCV 2020*, pp. 523–540, Cham, 2021b. Springer International Publishing. ISBN 978-3-030-69535-4.

Tingman Yan, Xilian Yang, Genke Yang, and Qunfei Zhao. Hierarchical belief propagation on image segmentation pyramid. *IEEE Transactions on Image Processing*, 32:4432–4442, 2023. doi: 10.1109/TIP.2023.3299192.

Jonathan S. Yedidia, William T. Freeman, and Yair Weiss. *Understanding Belief Propagation and Its Generalizations*, pp. 239–269. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003. ISBN 1558608117.

J.S. Yedidia, W.T. Freeman, and Yair Weiss. Constructing free-energy approximations and generalized belief propagation algorithms. *Information Theory, IEEE Transactions on*, 51:2282 – 2312, 08 2005. doi: 10.1109/TIT.2005.850085.

Elkafi Hassini Yunfei Ma, Amir Amiri and Saiedeh Razavi. Transportation data visualization with a focus on freight: a literature review. *Transportation Planning and Technology*, 45(4):358–401, 2022.

J. Zhang. The mean field theory in em procedures for blind markov random field image restoration. *IEEE Transactions on Image Processing*, 2(1):27–40, 1993. doi: 10.1109/83.210863.

Jun Zhang and G.G. Hanauer. The application of mean field theory to image motion estimation. *IEEE Transactions on Image Processing*, 4(1):19–33, 1995. doi: 10.1109/83.350816.

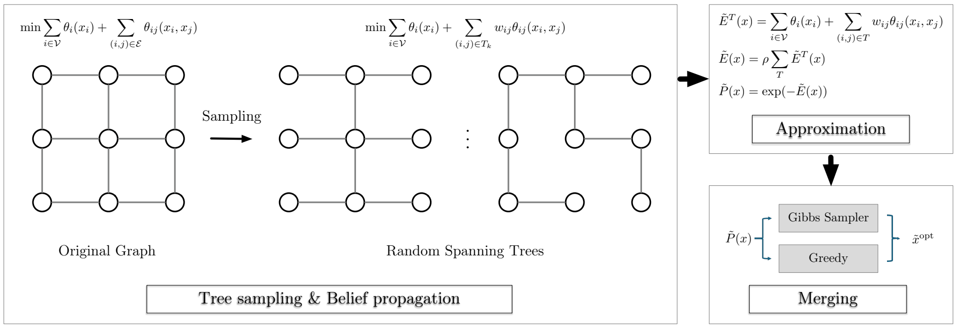

$\min \sum_{i \in \mathcal{V}} \theta_i(x_i) + \sum_{(i,j) \in \mathcal{E}} \theta_{ij}(x_i, x_j)$
$\min \sum_{i \in \mathcal{V}} \theta_i(x_i) + \sum_{(i,j) \in T_k} w_{ij}\theta_{ij}(x_i, x_j)$
Sampling
Original Graph
Random Spanning Trees
Tree sampling & Belief propagation
$\tilde{E}^T(x) = \sum_{i \in \mathcal{V}} \theta_i(x_i) + \sum_{(i,j) \in T} w_{ij}\theta_{ij}(x_i, x_j)$
$\tilde{E}(x) = \rho \sum_T \tilde{E}^T(x)$
$\tilde{P}(x) = \exp(-\tilde{E}(x))$
Approximation
$\tilde{P}(x)$
Gibbs Sampler
Greedy
$\tilde{x}^{\mathrm{opt}}$
Merging

Figure 3: Pipeline of the proposed approach.

## A Schematic diagram of SPT

Here we present the schematic diagram of the proposed approach in Fig. 3, anticipating to provide a more intuitive understanding.

## B Proof of Lemma 1

**Lemma 1.** *Given a uniform spanning tree distribution $\Omega(\mathcal{T})$ and the corresponding edge appearance probabilities $\{\rho_{ij} | \forall (i,j) \in \mathcal{E}\}$, when $|\mathcal{K}| = |\mathcal{T}|$, the approximation energy Eq. 9 and the original energy coincide.*

*Proof.* When $|\mathcal{K}| = |\mathcal{T}|$,

$$\begin{aligned}\tilde{E}(X) &= \sum_{i \in \mathcal{V}} \theta_i(x_i) + \sum_{(i,j) \in \mathcal{E}} |\{T_k | (i,j) \in T_k, \forall k \in \mathcal{K}\}| \frac{1}{\rho_{ij}} \theta_{ij}(x_i, x_j) \\ &= \sum_{i \in \mathcal{V}} \theta_i(x_i) + \sum_{(i,j) \in \mathcal{E}} \theta_{ij}(x_i, x_j),\end{aligned} \tag{17}$$

which precisely is the original energy. □

## C Proof of Lemma. 2

**Lemma 2.** *By applying the Monte Carlo Approximation, the expectation of the approximation is the original energy.*

$$\mathbb{E}_{T_k \sim \Omega(\mathcal{T})} \left[ \sum_{i \in \mathcal{X}} \theta_i(x_i) + \frac{1}{|\mathcal{K}|} \sum_{k \in \mathcal{K}} \sum_{(i,j) \in T_k} w_{ij} \theta_{i,j}(x_i, x_j) \right] = \sum_{i \in \mathcal{X}} \theta_i(x_i) + \sum_{(i,j) \in \mathcal{E}} \theta_{ij}(x_i, x_j) \tag{13}$$

*Proof.* Since the unary term $\theta_i(x_i)$ is included in all spanning trees, we only need to prove the pairwise term in expectation is identical to the original pairwise energy term.

$$\mathbb{E}_{T_k \sim \Omega(\mathcal{T})} \left[ \frac{1}{|\mathcal{K}|} \sum_{k=1}^{|\mathcal{K}|} \sum_{(i,j) \in \mathcal{T}_{||}} w_{ij} \theta_{ij}(x_i, x_j) \right] = \frac{1}{|\mathcal{K}|} \sum_{k=1}^{|\mathcal{K}|} \mathbb{E}_{T_k \sim \Omega(\mathcal{T})} \left[ \sum_{(i,j) \in \mathcal{T}_k} w_{ij} \theta_{ij}(x_i, x_j) \right] \tag{18}$$

Since each tree is sampling uniformly and independently

$$\frac{1}{|\mathcal{K}|}\sum_{k=1}^{|\mathcal{K}|}\mathbb{E}_{T_k\sim\Omega(\mathcal{T})}\left[\sum_{(i,j)\in\mathcal{T}_k} w_{ij}\theta_{ij}(x_i,x_j)\right] \tag{19}$$

$$=\frac{1}{|\mathcal{K}|}\sum_{k=1}^{|\mathcal{K}|}\mathbb{E}_{T}\left[\sum_{(i,j)\in\mathcal{T}_k} w_{ij}\theta_{ij}(x_i,x_j)\right] \tag{20}$$

$$=\mathbb{E}_{T}\left[\sum_{(i,j)\in\mathcal{T}_k} w_{ij}\theta_{ij}(x_i,x_j)\right] \tag{21}$$

Here we use $\mathbb{E}_T$ to denote $\mathbb{E}_{T\sim\text{uniform}(T)}$.

Introduce indicator variables:

$$I_{(i,j)}^{(k)}=\begin{cases}1 & \text{if edge } (i,j) \text{ is in spanning tree } T_k\\ 0 & \text{otherwise}\end{cases} \tag{22}$$

We have $\mathbb{E}[I_{(i,j)}^{(k)}]=\rho_{ij}$.

Then,

$$\mathbb{E}_{T}\left[\sum_{(i,j)\in\mathcal{T}_k} w_{ij}\theta_{ij}(x_i,x_j)\right] \tag{23}$$

$$=\mathbb{E}_{T}\left[\sum_{(i,j)\in\mathcal{E}} w_{ij}I_{(i,j)}^{T}\theta_{ij}(x_i,x_j)\right] \tag{24}$$

$$=\sum_{(i,j)\in\mathcal{E}} w_{ij}\mathbb{E}_T[I_{(i,j)}^{T}]\theta_{ij}(x_i,x_j) \tag{25}$$

$$=\sum_{(i,j)\in\mathcal{E}} \frac{1}{\rho_{ij}}\rho_{ij}\theta_{ij}(x_i,x_j) \tag{26}$$

$$=\sum_{(i,j)\in\mathcal{E}} \theta_{ij}(x_i,x_j) \tag{27}$$

Therefore, you could see that the expectation of the approximation is the original energy function.

□

## D Proof of Theorem 1

**Theorem 1.** *Given spanning tree distribution $\Omega(\mathcal{T})$ and the corresponding edge appearance probability $\{\rho_{ij}|\forall(i,j)\in\mathcal{E}\}$, the following error bound of the approximation energy Eq. equation 9 holds with probability at least $1-\delta$.*

$$|E(X)-\tilde{E}(X)|\leq\sqrt{\frac{1}{|\mathcal{K}|}\sum_{(i,j)\in\mathcal{E}}\theta_{ij}^2(x_i,x_j)(\frac{1-\rho_{ij}}{\rho_{ij}})}\frac{1}{\sqrt{\delta}}. \tag{14}$$

*When*

$$|\mathcal{K}|\geq\frac{1}{\delta\eta^2}\sum_{(i,j)\in\mathcal{E}}\theta_{ij}^2(x_i,x_j)(\frac{1-\rho_{ij}}{\rho_{ij}}), \tag{15}$$

*we have $P(|E(X)-\tilde{E}(X)|\geq\eta)\leq\delta$.*

*Proof.* Given

$$E(X)=\sum_{i\in\mathcal{X}}\theta_i(x_i)+\sum_{(i,j)\in\mathcal{E}}\theta_{i,j}(x_i,x_j), \tag{28}$$

$$\tilde{E}(X)=\sum_{i\in\mathcal{X}}\theta_i(x_i)+\frac{1}{|\mathcal{K}|}\sum_{k\in\mathcal{K}}\sum_{(i,j)\in\mathcal{E}}w_{ij}\theta_{i,j}(x_i,x_j), \tag{29}$$

We have

$$\Delta E(X) = E(X) - \tilde{E}(X) \tag{30}$$

$$= (\frac{1}{|\mathcal{K}|}\sum_{k\in\mathcal{K}}\sum_{(i,j)\in\mathcal{E}} w_{ij}\theta_{i,j}(x_i,x_j)) - \sum_{(i,j)}\theta_{i,j}(x_i,x_j). \tag{31}$$

Introduce indicator variables:

$$I_{(i,j)}^{(k)} = \begin{cases} 1 & \text{if edge } (i,j) \text{ is in spanning tree } T_k \\ 0 & \text{otherwise} \end{cases} \tag{32}$$

Then the error becomes:

$$\Delta E(X) = \sum_{(i,j)\in\mathcal{E}}((\frac{1}{|\mathcal{K}|}I_{(i,j)}^{(k)}w_{ij}\theta_{ij}(x_i,x_j)) - \theta_{ij}(x_i,x_j)). \tag{33}$$

Now define the Per-Edge error $\Delta E_{ij}$. Then

$$\Delta E(X) = \sum_{(i,j)\in\mathcal{E}}\Delta E_{ij}. \tag{34}$$

Define random variable $Z_{ij}^{(k)} = (I_{(i,j)}^{(k)}w_{ij} - 1)\theta_{ij}(x_i,x_j)$. Then,

$$\Delta E_{ij} = \frac{1}{|\mathcal{K}|}\sum_{k\in|\mathcal{K}|}\Delta Z_{ij}^{(k)}. \tag{35}$$

Since $\mathbb{E}[I_{ij}^{(k)}] = \rho_{ij}$

$$\mathbb{E}[Z_{ij}^{(k)}] = (\mathbb{E}[I_{ij}^{(k)}w_{ij}] - 1\theta_{ij}(x_i,x_j) \tag{36}$$

$$= (\rho_{ij}w_{ij} - 1)\theta_{ij}(x_i,x_j) \tag{37}$$

$$= 0. \tag{38}$$

Then,

$$Var[Z_{ij}^{(k)}] = (w_{ij}\theta_{ij}(x_i,x_j))^2\rho_{ij}(1-\rho_{ij}) \tag{39}$$

$$Var[Z_{ij}^{(k)}] = (\frac{1}{\rho_{ij}}\theta_{ij}(x_i,x_j))^2\rho_{ij}(1-\rho_{ij}) \tag{40}$$

$$Var[\Delta E_{ij}] = \frac{Var[Z_{ij}^{(k)}]}{|\mathcal{K}|} \tag{41}$$

$$Var[\Delta E(X)] = \sum_{(i,j)\in\mathcal{E}} Var[\Delta E_{ij}]. \tag{42}$$

By Chebyshev's Inequality,

$$P(|\Delta E(X)| \geq \eta) \leq \frac{Var[\Delta E(X)]}{\eta^2} \tag{43}$$

$$\leq \frac{1}{|\mathcal{K}|\eta^2}\sum_{(i,j)\in\mathcal{E}}\theta_{ij}^2(x_i,x_j)\frac{(1-\rho_{ij})}{\rho_{ij}}. \tag{44}$$

To ensure that the probability of the error exceeding $\eta$, is less than or equal to $\delta$,

$$\frac{1}{|\mathcal{K}|\eta^2}\sum_{(i,j)\in\mathcal{E}}\theta_{ij}^2(x_i,x_j)\frac{(1-\rho_{ij})}{\rho_{ij}} \leq \delta. \tag{45}$$

Solving for $|\mathcal{K}|$,

$$|\mathcal{K}| \geq \frac{1}{\delta\eta^2}\sum_{(i,j)\in\mathcal{E}}\theta_{ij}^2(x_i,x_j)\frac{(1-\rho_{ij})}{\rho_{ij}}. \tag{46}$$

Then, the error bound for the adjusted energy approximation E(X )is,

$$|\Delta E(X)| \leq \sqrt{\frac{1}{|\mathcal{K}|} \sum_{(i,j)\in\mathcal{E}} \theta_{ij}^2(x_i, x_j) \frac{(1-\rho_{ij})}{\rho_{ij}}} \cdot \frac{1}{\sqrt{\delta}}. \tag{47}$$

This bound holds with probability at least $1 - \delta$. □

# E Complexity Analysis

The algorithm consists of three parts: calculating the effective resistance, sampling spanning trees, and applying belief propagation on the spanning trees. Generally, the time complexity of computing the effective resistance is $O(MN^3)$. For sparse graphs, the factor $M$ can be considered a negligible coefficient. However, in scenarios involving large and complex graphs, the computational complexity of calculating the probability matrix becomes dominant relative to other operations. This computational burden represents the primary limitation for the broader application of our proposed method.

For the second step, the time complexity depends on the chosen spanning algorithm. In our approach, we adopt the method proposed in (Wilson, 1996), which has a runtime of $O(N^3)$. As discussed in Section. 4.2, Depth-First Search (DFS) was proposed as a potential approach to enhance computational efficiency at this stage. However, empirical results indicate that the performance difference between these two sampling methods is relatively modest in practice. This can be attributed to the availability of optimization techniques and the fundamental similarity between random walk-based sampling and DFS-based approaches.

The time complexity of the belief propagation is determined by the size of the spanning tree, which remains fixed when the graph is constant. Given $|\mathcal{E}_{\text{tree}}| = N - 1$, the time complexity becomes $O(N)$. It is worth noting that our algorithm converges rapidly, typically concluding within fewer than 10 iterations in most cases. Assume the average number of dependencies of each node is $b$, the time complexity of gibbs sampler would be $O(iter \times Nk)$. This efficiency allows us to disregard the term $iter$ without significantly affecting the complexity of the algorithm. As a result, the overall time complexity of the algorithm is $O(MN^3 + N)$.

# F Effective Resistance Calculation

The calculation of resistance distance generally follows the procedure outlined in Theorem 2. The main contributor to the time complexity of this part is the computation of the Moore-Penrose inverse, as we need to use the effective resistance on all the edges of the graph.

**Theorem 2** (**Theorem 2.7 in** (Vos, 2016))**.** *The effective resistance between a pair of vertices $(i, j)$ is defined as Reff$_{i,j} \coloneqq \Gamma_{i,i} + \Gamma_{j,j} - \Gamma_{i,j} - \Gamma_{j,i}$, where $\Gamma = (L + \frac{1}{|\mathcal{V}|}\Phi)^\dagger$, with $^\dagger$ denotes the Moore-Penrose inverse, $L$ the Laplacian matrix of $\mathcal{G}$, $|\mathcal{V}|$ is the number of vertices in $\mathcal{G}$, and $\Phi$ is the $|\mathcal{V}| \times |\mathcal{V}|$ matrix contain all 1s. $\Gamma_{i,j}$ is the $(i, j)$ entry of the Moore-Penrose inverse of the Laplacian matrix.*

# G Comparison to Tree Sampling algorithm

As previously discussed, the Tree Sampling algorithm proposed by Hamze & de Freitas (2004) is applicable only to graphs with specific structures that can be divided into two cycle-free parts. Under these conditions, the algorithm focuses on solving the MAP estimation problems on grid graphs with observation nodes. The MAP estimate is given by:

$$\min_X E(X) = \min_{\{x_i\},\forall i\in\mathcal{V}} \left\{ \sum_{i\in\mathcal{V}} \theta_i(x_i, y_i) + \sum_{(i,j)\in\mathcal{E}} \theta_{ij}(x_i, x_j) \right\}. \tag{48}$$

here $y_i$ is the observation of the node $i$ which is deterministic value and $y_i, i \in \mathcal{V}$ has the same value range as the $x_i, i \in \mathcal{V}$.

To comprehensively evaluate the performance of our Spanning Tree algorithm and the Tree Sampling algorithm, we conducted experiments using grid graphs of varying sizes: 10×10 grids with 100 nodes, 20×20 grids with 400 nodes, and 30×30 grids with 900 nodes. Each node in these grids was assigned one of 10 possible labels. The pairwise energy and the energy between variables and their observations were defined using a squared label difference function. Observations were generated randomly for each instances.

For each grid size, we repeated the experiment across 10 different random instances to ensure robustness and reliability of the results. Each of these two algorithms run for 20 iterations. The number of spanning trees used by SPT is 10. As illustrated in Fig. 4, our algorithm consistently outperformed the Tree Sampling algorithm across all grid sizes. Importantly, the performance gap between the two algorithms remained consistent as the grid size increased.

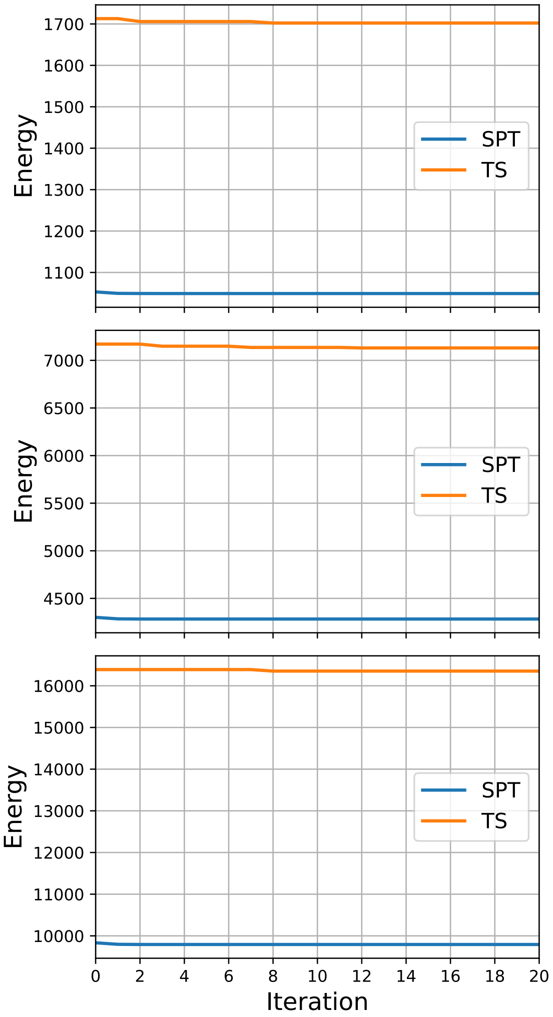


Figure 4: Results of SPT and TS on synthetic grid problems with squared label difference. The horizontal and vertical axes in the figure correspond to #number of iterations and average value of energy $E(x|\tilde{\theta}, y)$, respectively. From top to bottom, the grid sizes are $10\times10$, $20\times20$, $30\times30$.

## H Generation of different sub-graph structures

In our previous discussions, we proposed using cycle-free structures to decompose the original graph, allowing us to leverage problems that can be solved exactly as approximations for the original problem. In addition to the advantages we outlined earlier, the spanning trees have demonstrated an extraordinary ability to provide more accurate approximations. We have also explored other structural alternatives, including chains, random trees (not necessarily spanning trees), and **random structures** generated using random walks. To obtain the **chains**, we used the following procedure: we would start at a randomly selected node and then perform a depth-first search to traverse the graph and collect the subsequent nodes. When sampling **random trees**, we would stop the sampling procedure once the number of nodes in the generated tree reached our target threshold. In this process, we allowed an agent to walk along the edges of the graph for a certain number of steps. After the agent stopped, the nodes it had visited and the edges it had traversed were extracted to form the sub-graph.

Table 3: SPT inference optimality gaps with different number of trees on trivial MRF instances. Numbers are obtained optimality gaps.

| Graph | #Nodes/#Edges | 10 trees | 20 trees | 30 trees | 40 trees | 50 trees | 60 trees | 70 trees | 80 trees | 90 trees | 100 trees | OPT energy |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| INSTANCE_1 | 64/143 | 5 | 6 | 3 | 2 | 2 | 3 | 2 | 2 | 3 | 2 | 94 |
| INSTANCE_2 | 64/156 | 7 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 134 |
| INSTANCE_3 | 64/124 | 1 | 2 | 2 | 2 | 2 | 2 | 2 | 1 | 1 | 1 | 104 |
| INSTANCE_4 | 64/116 | 3 | 4 | 5 | 6 | 5 | 3 | 3 | 3 | 3 | 2 | 131 |
| INSTANCE_5 | 64/133 | 1 | 1 | 2 | 2 | 2 | 1 | 2 | 2 | 2 | 1 | 117 |

## I Error Gap on trivial MRFs

We conducted additional experiments on MRFs with 64 nodes and increased the number of trees used to 100. We use the optimality gap as the evaluation standard. The results are listed in the Table 3. You can see that as more trees are sampled, the inference results converge to a point with only occasional fluctuations, and the absolute fluctuation is 1. This could be caused by the problem structure where some variable values offer similar energies that are difficult to distinguish, which is a characteristic of BP-based methods. Since our method is also based on BP, it is inevitable that we encounter the same issue. The calculation of Optimality gap is shown in Eq. 49 which could be found in the user manual of Mosek (ApS, 2025).

$$\text{Optimality Gap} = |\tilde{E}(X) - E^*(X)| \tag{49}$$

## J Example of Energy Weight Adjustment

In this section we use a toy example to show the importance of weight adjustment when using uniform spanning trees to decompose the original problem. In Fig. 5, we show a graph with four nodes and four edges, and we show all of the three spanning trees of it. Except edge $(1, 3)$, the probabilities of the other edges appear in a uniform spanning tree are all $\frac{2}{3}$, which means the weights of the pairwise energies on these edges are all $\frac{3}{2}$. The probability of each of the trees being sampled is $\frac{1}{3}$. In Fig. 5 we using color red to denote the edges that have probability of $\frac{1}{3}$ being selected and the color blue to denote the edge that appear in all the uniform spanning trees.

Without loss of generality, we could assume the unary energies are all zero. Then we define the pairwise energies as follows. Each random variable has 2 possible states $\{0, 1\}$.

| $x_1$<br>$x_0$ | 0 | 1 |
|---|---|---|
| 0 | 1 | 0 |
| 1 | 0 | 0 |

| $x_2$<br>$x_0$ | 0 | 1 |
|---|---|---|
| 0 | 2 | 0 |
| 1 | 0 | 0 |

| $x_2$<br>$x_1$ | 0 | 1 |
|---|---|---|
| 0 | 3 | 0 |
| 1 | 0 | 0 |

| $x_3$<br>$x_1$ | 0 | 1 |
|---|---|---|
| 0 | 4 | 0 |
| 1 | 0 | 0 |

Now we calculate the energy when all the variables are at state 0. The original energy is $E(0, 0, 0, 0) = 1+2+3+4 = 10$. If without weight, the energies on these trees would be $E_{T_1}(0, 0, 0, 0) = 8$, $E_{T_2}(0, 0, 0, 0) = 7$, $E_{T_3}(0, 0, 0, 0) = 8$.

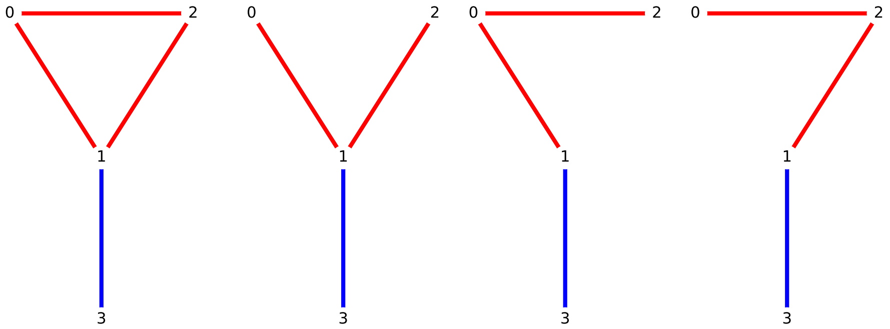
Then merging them together by

$$\tilde{E}(x) = \sum_{k \in \mathcal{K}} \rho_{T_k} \sum_{(i,j) \in T_k} \theta_{ij}(x_i, x_j) \tag{50}$$

$$\tag{51}$$

we can get the approximation value is $\frac{23}{3}$, which deviates from the original energy.

If we we adjust the pairwise energy using $1/\rho_{ij}$. the energies on these trees would be $E_{T_1}(0, 0, 0, 0) = \frac{3}{2}(1 + 3) + 4 = 10$, $E_{T_2}(0, 0, 0, 0) = \frac{3}{2}(1 + 2) + 4 = 8.5$, $E_{T_3}(0, 0, 0, 0) = \frac{3}{2}(2 + 3) + 4 = 11.5$. Then merging by Eq .equation 9, the approximation is $\frac{1}{3}(10 + 8.5 + 11.5) = 30$ which is exactly the original energy.



(a) Original Graph (b) Spanning Tree 1 (c) Spanning Tree 2 (d) Spanning Tree 3

Figure 5: Schematic diagrams of spanning trees of a graph with cycle.

Table 4: Inference time on PCI instances.

| GRAPH | #VAR/#CON | #NODES/#EDGES | LBP | TRBP | SPT |
|---|---|---|---|---|---|
| PCI_INSTANCE_1 | 955/2496 | 30/165 | 0.359283s | 0.441703s | 0.405572s |
| PCI_INSTANCE_2 | 1588/4409 | 40/311 | 0.39226s | 0.652174s | 0.43437s |
| PCI_INSTANCE_3 | 17684/52673 | 80/1522 | 0.853267s | 0.91823s | 0.770843s |
| PCI_INSTANCE_4 | 65713/193287 | 286/10565 | 2.14694s | 6.211s | 3.609s |

## K INFERENCE TIME ON PCI INSTANCES

In Table 4, we present the inference times of LBP, TRBP, and our SPT algorithm on the real PCI instances used in Section 5.3. Notably, despite SPT's $O(N^3)$ time complexity, its inference times on these instances fall between those of LBP and TRBP.

## L TRANSFORMING MIP PROBLEMS OF PCI INTO MRF PROBLEMS

The Mixed Integer Programming(MIP) format of PCI problems is as follows:

$$\min_{z,L} \quad \sum_{(i,j)\in\mathcal{E}} a_{ij} L_{ij} \tag{52}$$

$$\text{s.t.} \quad z_{np} \in \{0,1\}, \quad \forall n \in N, p \in P \tag{53}$$

$$\sum_{p\in P} z_{np} = 1, \quad \forall n \in N. \tag{54}$$

$$\sum_{p\in M_{ih}} z_{n_i p} + \sum_{p\in M_{jh}} z_{n_j p} - 1 \leq L_{ij}, \forall (i,j) \in \mathcal{E}, \forall h \in \{0,1,2\}. \tag{55}$$

where $n$ is the index for devices, and $N$ is the set of these indices. $P$ stands for the possible states of each device. $M_{ih}$ stands for the possible states set for node $n_i$. $L_{ij}$ is the cost when given a certain choices of the states of device $i$ and device $j$, $a_{ij}$ is the coefficient of the cost in the objective function. There is an $(i,j) \in \mathcal{E}$ means there exists interference between these two devices. In the MIP formulation of the PCI problems, there are three types of constraints. Combining equation 53 and equation 54 together implies that each device must select one state, and only one state can be chosen at a given time. The constraint equation 55 indicates that interference occurs between two devices only if they choose specific states. The impact on the entire system is determined by the corresponding value of $L_{ij}$ and its coefficient. Since interference always exists, the objective is to minimize its degree.

To transform these problems into MRF problems, we can use equation 54 to represent nodes, where each equation 53 corresponds to the discrete states of a given node. Since only one state can be chosen at a time, the constraints equation 53 and equation 54 are naturally satisfied. By processing equation 55, we identify the edges and their associated energies. If we find $z_{n_i p}$ and $z_{n_j p}$ in the same constraint from equation 55, we can formulate an edge $(i,j)$. By selecting different values for $z_{n_i p}$ and $z_{n_j p}$, we can determine the minimum value of $L_{ij}$ that satisfies the constraint. The product of $L_{ij}$ and $a_{ij}$ represents the energy for the edge $(i,j)$ under the combination of these two states. When all the states of the nodes are fixed, the values of the edge costs become fixed as well. This implies that the objective function is the summation of all the edge energies. Since the PCI problems do not include unary terms, we will neglect them during the transformation process.

**Example**
The original problem is

$$\begin{aligned}
\min_{z,L} \quad & L_{1,2} + 2L_{2,3} \\
\text{s.t.} \quad & z_{np} \in \{0,1\}, && \forall n \in \{1,2,3\}, p \in \{1,2,3\} \\
& \sum_{p \in P} z_{np} = 1, && \forall n \in \{1,2,3\}. \\
& z_{11} + z_{21} - 1 \leq L_{1,2} \\
& z_{13} + z_{22} - 1 \leq L_{1,2} \\
& z_{12} + z_{23} - 1 \leq L_{1,2} \\
& z_{21} + z_{31} - 1 \leq L_{2,3} \\
& z_{22} + z_{32} - 1 \leq L_{2,3} \\
& z_{23} + z_{33} - 1 \leq L_{2,3}
\end{aligned} \tag{56}$$

Then the corresponding MRF problem is

$$\min \theta_{1,2}(x_1, x_2) + \theta_{2,3}(x_2, x_3) \tag{57}$$

the energy on edge $(x_1, x_2)$ and edge $(x_2, x_3)$ are as follows:

| $x_2$<br>$x_1$ | $z_{21}$ | $z_{22}$ | $z_{23}$ |
|---|---|---|---|
| $z_{11}$ | 1 | 0 | 0 |
| $z_{12}$ | 0 | 0 | 1 |
| $z_{13}$ | 0 | 1 | 0 |

| $x_3$<br>$x_2$ | $z_{31}$ | $z_{32}$ | $z_{33}$ |
|---|---|---|---|
| $z_{21}$ | 2 | 0 | 0 |
| $z_{22}$ | 0 | 2 | 0 |
| $z_{23}$ | 0 | 0 | 2 |

## M GENERATION OF LOCALLY AND SPARSELY CONNECTED MRFS

We begin by uniformly placing the nodes in a 2-dimensional space. Next, we iteratively traverse all the nodes in an arbitrary order. For each node visited, we identify its k-nearest neighbors and create edges between them until the required average degree is achieved. Once all nodes have been visited, the graph construction is complete.

## N PARALLELIZATION

When the size of the graph increases, our algorithms still function, but they become time-consuming. This is because the time complexity, as we have just analyzed, is highly dependent on both $N$ and $M$ (where $M$ is related to the number of states $n$ in the Markov chain). Evidently, acquiring comprehensive information about the entire graph would require more than one tree. Such a requirement could result in a notable slowdown of our algorithm due to the necessity of performing belief propagation on each tree. Nonetheless, the algorithm we have introduced can be seamlessly modified for parallel processing. Once the edge selection probabilities have been calculated, both the spanning tree sampling process and the subsequent belief propagation on each sampled tree can be performed independently and in parallel. This adaptability enables us to significantly improve efficiency.

## O ITERATION WITH DIFFERENT BATCHES OF SPANNING TREES

Besides using fixed trees to perform the estimation, we could also generate another batch of trees to conduct the estimation in a new iteration, similar to what was done when using different subgraph structures for the estimation. However, this could be an expensive strategy to adopt. Note the analysis in **Appendix E**, where the time complexity of sampling a spanning tree is $O(MN^3 + N)$, which could incur a significant computational cost as the graph size increases.

To get a comprehensive understanding of whether we could gain more by spending more time on sampling trees and how the two strategies would perform with a similar time budget, we experimented on 40x40 grid graphs with 1600 nodes. Each random node has 10 to 16 possible states to choose from, the pairwise energy is determined by a square function, and the unary energy is set to zero. For the SPT method using fixed trees, the number of trees is set to 20. For the SPT with tree resampling, the number of trees is varied as {1, 5, 10, 15, 20}. All the SPT variants are run for 20 iterations.

As shown in Fig. 6, energy curves for each setting of the SPT methods reveal that both the SPT variants with 20 trees achieve the best and identical results. As the number of trees used per iteration increases, the gain on the final energy decreases, and the improvement rate diminishes quickly. However, the actual time cost of the SPT with fixed trees lies between the SPT with tree resampling using 1 tree and 5 trees and far less than re-sampling 20 trees at each iteration. This suggests that, although we could resample the trees during the iterative process, the additional time spent on this may not be worthwhile.

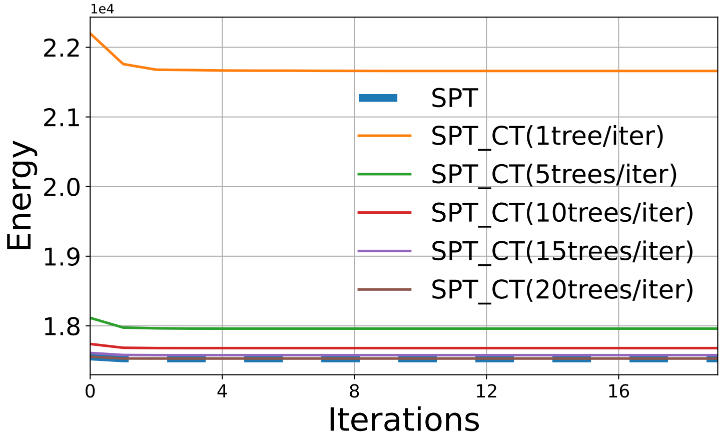
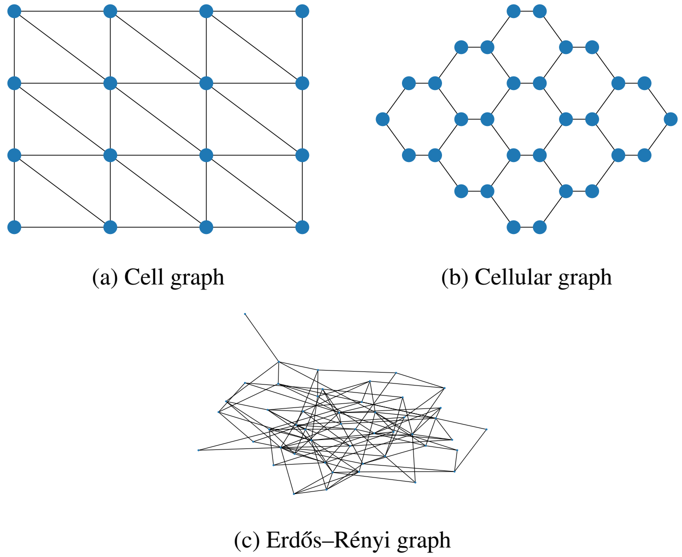


Figure 6: Energy curves of SPT using fixed trees or resample spanning trees at each iteration.

# P Instance Topology

## P.1 Synthetic Problems

Fig. 7 illustrates schematic diagrams of cellular ,cell graph and Erdős–Rényi graph structures. And in Fig. 8 illustrates two schematic diagrams of the locally and sparsely connected graphs we generated. Note they do not correspond to any testing instances.



(a) Cell graph

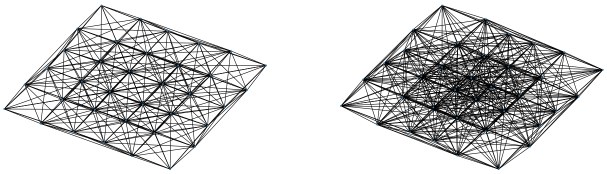
(b) Cellular graph

(c) Erdős–Rényi graph

Figure 7: Schematic diagrams of cell graph, cellular graph and Erdős–Rényi graph.

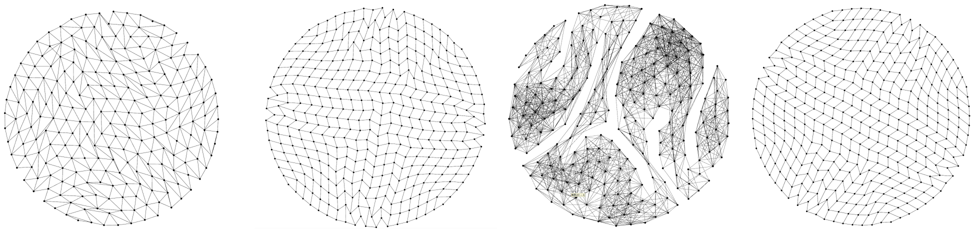
## P.2 UAI Inference Competition

We visualize the topology of four instances from UAI competition in Fig. 9.



(a) Average degree 17 (b) Average degree 24

Figure 8: Schematic diagrams of locally and sparsely connected graphs.

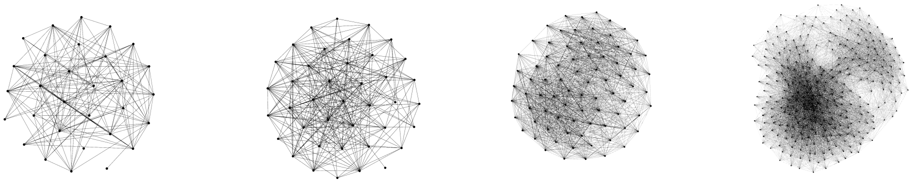


(a) Segmentation_14 (b) ProteinFolding_11 (c) ProteinFolding_12 (d) Grids_30

Figure 9: Schematic diagrams of some UAI inference competition instances. Zoom in for better view.

### P.3 PCI Instances

All the topologies of the PCI instances are in Fig. 10.



(a) PCI_INSTANCE_1 (b) PCI_INSTANCE_2 (c) PCI_INSTANCE_3 (d) PCI_INSTANCE_4

Figure 10: Schematic diagrams of PCI instances. Zoom in for better view.

## Q LLM statement

In our research process, we utilized large language models primarily to support technical writing aspects rather than for generating research content. These tools were employed specifically for grammar checking, spell correction, improving sentence structure, and enhancing the overall readability of the manuscript. All scientific contributions, technical analyses, experimental designs, and conclusions presented in this paper are the original work of the authors, with language models serving only as writing assistance tools.