Local approximate inference algorithms

Kyomin Jung and Devavrat Shah

***Abstract*—We present a new local approximation algorithm for computing Maximum a Posteriori (MAP) and log-partition function for arbitrary exponential family distribution represented by a finite-valued pair-wise Markov random field (MRF), say G. Our algorithm is based on decomposition of G into *appropriately* chosen small components; then computing estimates locally in each of these components and then producing a *good* global solution. Our algorithm for log-partition function provides provable upper and lower bounds on the correct value for arbitrary graph G. For MAP, our algorithm provides approximation with quantifiable error for arbitrary G. Specifically, we show that if the underlying graph G either excludes some finite-sized graph as its minor (e.g. Planar graph) or has low doubling dimension (e.g. any graph with *geometry*), then our algorithm will produce solution for both questions within *arbitrary accuracy*. The running time of the algorithm is $\Theta(n)$ (n is the number of nodes in G), with constant dependent on accuracy and either doubling dimension, or maximum vertex degree and the size of the graph that is excluded as a minor (e.g. 3 for all Planar graphs).**

We present a message-passing implementation of our algorithm for MAP computation using self-avoiding walk of graph. In order to evaluate the computational cost of this implementation, we derive novel tight bounds on the size of self-avoiding walk tree for arbitrary graph, which may be of interest in its own right.

As a consequence of our algorithmic result, we show that the normalized log-partition function (also known as free-energy) for a class of *regular* MRFs (e.g. Ising model on 2-dimensional grid) will converge to a limit, that is computable to an arbitrary accuracy, as the size of the MRF goes to infinity. This method, like classical sub-additivity method, is likely to be widely applicable.

***Index Terms*—Markov random fields; approximate inference; low doubling-dimension graphs; minor-excluded graphs; planar graphs; MAP-estimation; log-partition function; message-passing algorithms; self-avoiding walk.**

I. Introduction

Markov Random Field (MRF) [1] based exponential family of distribution allows for representing distributions in an intuitive parametric form. Therefore, it has been successful in modeling many applications (see, [2] for details). The key operational questions of interest are related to statistical inference: computing most likely assignment of (partially) unknown variables given some observations and computation of probability of an assignment given the partial observations (equivalently, computing log-partition function). In this paper, we study the question of designing efficient local algorithms for solving these inference problems.

A. Previous work

The question of finding MAP (or ground state) of a given MRF comes up in many important application areas such as coding theory, discrete optimization, image denoising. Similarly, log-partition function is used in counting combinatorial objects [3], loss-probability computation in computer networks, [4], etc. Both problems are NP-hard for exact and even (constant) approximate computation for arbitrary graph G. However, the above stated applications require solving these problems using very simple algorithms. A popular successful approach for designing efficient heuristics has been as follows. First, identify a wide class of graphs that have simple algorithms for computing MAP and log-partition function. Then, for any given graph, approximately compute solution either by using that simple algorithm as a heuristic or in a more sophisticated case, by possibly solving multiple sub-problems induced by sub-graphs with good graph structures and then combining the results from these sub-problems to obtain a global solution.

Such an approach has resulted in many interesting recent results starting the Belief Propagation (BP) algorithm designed for Tree graph [1]. Since there is a vast literature on this topic, we will recall only few results. In our opinion, two important algorithms proposed along these lines of thought are the generalized belief propagation (BP) [5] and the tree-reweighted algorithm (TRW) [6]–[8]. Key properties of interest for these iterative procedures are the correctness of their fixed points and convergence. Many results characterizing properties of the fixed points are known starting from [5]. Various sufficient conditions for their convergence are known starting [9]. However, simultaneous convergence and correctness of such algorithms are established for only specific problems, e.g. [10]–[12].

Finally, we discuss two relevant results. The first result is about properties of TRW. The TRW algorithm provides provable upper bound on log-partition function for arbitrary graph [8]. However, to the best of authors' knowledge the error is not quantified. The TRW for MAP estimation has a strong connection to specific Linear Programming (LP) relaxation of the problem [7]. This was made precise in a sequence of work by Kolmogorov [13], Kolmogorov and Wainwright [12] for binary MRF. It is worth noting that LP relaxation can be poor even for simple problems.

The second is an approximation algorithm proposed by Globerson and Jaakkola [14] to compute log-partition function using Planar graph decomposition (PDC). PDC uses techniques of [8] in conjunction with known result about exact computation of partition function for binary MRF when G is Planar and the exponential family has a specific form (binary pairwise and multiplicative potentials). Their algorithm

K. Jung is with the Department of Mathematics and D. Shah is with the Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, Cambridge, MA 02139 USA. Email: {kmjung,devavrat}@mit.edu

This work was supported in part by a Samsung graduate fellowship, NSF CAREER grant and DARPA ITMANET grant.

provides provable upper bound for arbitrary graph. However, they do not quantify the error incurred. Further, their algorithm is limited to binary MRF.

B. Contributions

We propose a novel local algorithm for approximate computation of MAP and log-partition function. For any $\varepsilon > 0$, our algorithm can produce an ε-approximate solution for MAP and log-partition function for *arbitrary* MRF G as long as G has either of these two properties: (a) G has low doubling dimension (see Theorems 2 and 5), or (b) G excludes a finite-sized graph as a minor (see Theorems 3 and 6). For example, Planar graph excludes $K_{3,3}, K_5$ as a minor and thus our algorithm provides approximation algorithms for Planar graphs.

The running time of the algorithm is $\Theta(n)$, with constant dependent on ε and (a) doubling dimension for doubling dimension graph, or (b) maximum vertex degree and size of the graph that is excluded as minor for minor-excluded graphs. For example, for 2-dimensional grid graph, which has doubling dimension $O(1)$, the algorithm takes $C(\varepsilon)n$ time, where $\log\log C(\varepsilon) = O(1/\varepsilon)$. On the other hand, for a planar graph with maximum vertex degree a constant, i.e. $O(1)$, the algorithm takes $C'(\varepsilon)n$ time, with $\log\log C'(\varepsilon) = O(1/\varepsilon)$.

In general, our algorithm works for any G and we can quantify bound on the error incurred by our algorithm. It is worth noting that our algorithm provides a provable lower bound on log-partition function as well unlike many of the previous results.

Our algorithm is primarily based on the following idea: First, decompose G into small-size connected components say $G_1, \dots, G_k$ by removing few edges of G. Second, compute estimates (either MAP or log-partition) in each of the G_i separately. Third, combine these estimates to produce a global estimate while *taking care* of the *error* induced by the removed edges. We show that the error in the estimate depends only on the edges removed. This error bound characterization is applicable for arbitrary graph.

For obtaining sharp error bounds, we need good graph decomposition schemes. Specifically, we use a new, simple and very intuitive randomized decomposition scheme for graphs with low doubling dimensions. For minor-excluded graphs, we use a simple scheme based on work by Klein, Plotkin and Rao [15] and Rao [16] that they had introduced to study the gap between max-flow and min-cut for multicommodity flows. In general, as long as G allows for such good edge-set for decomposing G into small components, our algorithm will provide a good estimate.

To compute estimates in individual components, we use dynamic programming. Since each component is small, it is not computationally burdensome. However, one may obtain further simpler heuristics by replacing dynamic programming by other method such as BP or TRW for computation in the components.

In order to implement dynamic programing using message-passing approach, we use construction based on self-avoiding walk tree. Self-avoiding walk trees have been of interest in statistical physics for various reasons (see book by Madras and Slade [17]). Recently, Weitz [18] obtained a surprising result that connected computation of marginal probability of a node in any binary MRF to that of marginal probability of a root node in an appropriate self-avoiding walk tree. We use a direct adaption of this result for computing MAP estimate to design message passing scheme for MAP computation. In order to evaluate computation cost, we needed tight bound on the size on self-avoiding walk tree of arbitrary graph G. We obtain a novel characterization of size of self-avoiding walk tree within a factor 8 for arbitrary graph G. This result should be of interest in its own right.

Finally, as a (somewhat unexpected) consequence of these algorithmic results, we obtain a method to establish existence of asymptotic limits of free energy for a class of MRF. Specifically, we show that if the MRF is d-dimensional grid and all node, edge potential functions are identical then the free-energy (i.e. normalized log-partition function) converges to a limit as the size of the grid grows to infinity. In general, such approach is likely to extend for any *regular enough* MRF for proving existence of such limit: for example, the result will immediately extend to the case when the requirement of node, edge potential being exactly the same is replaced by the requirement of they being chosen from a common distribution in an i.i.d. fashion.

C. Outline

The paper is organized as follows. Section II presents necessary background on graphs, Markov random fields, exponential family of distribution, MAP estimation and log-partition function computation.

Section III presents graph decomposition schemes. These decomposition schemes are used later by approximation algorithms. We present simple, intuitive and $O(n)$ running time decomposition schemes for graphs with low doubling dimension and graphs that exclude finite size graph as a minor. Both of these schemes are randomized. The first scheme is our original contribution. The second scheme was proposed by Klein, Plotkin and Rao [15] and Rao [16].

Section IV presents the approximation algorithm for computing log-partition function. We describe how it provides upper and lower bound on log-partition function for arbitrary graph. Then we specialize the result for two graphs of interest: low doubling dimension and minor excluded graphs.

Section V presents the approximation algorithm for MAP estimation. We describe how it provides approximate estimate for arbitrary graph with quantifiable approximation error. Then we specialize the result for two graphs of interest: low doubling dimension and minor excluded graphs.

Section VI describes message passing implementation of the MAP estimation algorithm for binary pair-wise MRF for arbitrary G. This can be used by our approximation algorithm to obtain message passing implementation. This algorithm builds upon work by Weitz [18]. We describe a novel tight bound on the size of self-avoiding walk tree for any G. This helps in evaluating the computation time. The message passing implementation has similar computation complexity as the centralized algorithm.

Section VII presents an experimental evaluation of our algorithm for popular synthetic model on a grid graph. We compare our algorithm with TRW and PDC algorithms and show that our algorithm is very competent. An important feature of our algorithm is scalability.

Section VIII presents the implication of our algorithmic result in establishing asymptotic limit of free energy for regular MRFs, such as an Ising model on d-dimensional grid.

D. How to read this paper: a suggestion

A reader, interested in obtaining a quick understanding of the results, should skip everything in Section III other than the definition of (ε, Δ) decomposition, and skip the Section VI completely. Reading these two sections at the very end may be helpful to parse the results with ease for all the readers.

II. Preliminaries

This section provides the background necessary for subsequent sections. We begin with an overview of some graph theoretic basics. We then describe formalism of Markov random field and exponential family of distribution. We formulate the problem of log-partition function computation and MAP estimation for Markov random field. We conclude by stating precise definitions of approximate MAP estimation and approximate log-partition function computation.

A. Graphs

An undirected graph $G = (V, E)$ consists of a set of vertices $V = \{1, \dots, n\}$ that are connected by set of edges $E \subset V \times V$. We consider only simple graphs, that is multiple edges between a pair of nodes or self-loops are not allowed. Let $\Gamma(v) = \{u \in V : (u, v) \in E\}$ denote the set of all neighboring nodes of $v \in V$. The size of the set $\Gamma(v)$ is the *degree* of node v, denoted as d_v. Let $d^* = \max_{v \in V} d_v$ be the maximum vertex degree in G. A *clique* of the graph G is a fully-connected subset C of the vertex set (i.e. $(u, v) \in E$ for all $u, v \in C$). Nodes u and v are called connected if there exists a path in G starting from u and ending at v or vice versa since G is undirected. Each graph G naturally decomposes into disjoint sets of vertices $V_1, \dots, V_k$ where for $1 \leq i \leq k$, any two nodes say $u, v \in V_i$ are connected. The sets $V_1, \dots, V_k$ are called the connected components of G.

We introduce a popular notion of *dimension* for graph G (see recent works [19] [20] [21] for relevant details). Define $\mathbf{d_G} : V \times V \to \mathbb{R}_+$ as

$$\mathbf{d_G}(i, j) = \text{length of the shortest path between } i \text{ and } j.$$

If $i = j$ then $\mathbf{d_G}(i, j) = 0$ and if i, j are not connected, then define $\mathbf{d_G}(i, j) = \infty$. It is easy to check that thus defined $\mathbf{d_G}$ is a metric on vertex set V. Define ball of radius $r \in \mathbb{R}_+$ around $v \in V$ as $\mathbf{B}(v, r) = \{u \in V : \mathbf{d_G}(u, v) < r\}$. Define

$$\begin{aligned}\rho(v, r) \quad = \quad & \inf\{K \in \mathbb{N} : \exists\ u_1, \dots, u_K \in V, \\ & \mathbf{B}(v, r) \subset \cup_{i=1}^{K} \mathbf{B}(u_i, r/2)\}.\end{aligned}$$

Then, $\rho(G) = \sup_{v \in V, r \in \mathbb{R}_+} \rho(v, r)$ is called the *doubling dimension* of graph G. Intuitively, this definition captures the notion of dimension d in the Euclidian space $\mathbb{R}^d$. It follows from definition that for any graph G, $\rho(G) = O(\log_2 n)$. We note the following property whose proof is presented in Appendix A.

Lemma 1: For any $v \in V$ and $r \in \mathbb{N}$, $|\mathbf{B}(v, 2^r)| \leq 2^{r\rho(\mathcal{M})}$.

Next, we introduce a class of graphs known as *minor-excluded* graphs (see a series of publications by Roberston and Seymour under ”the graph minor theory” project [22]). A graph H is called minor of G if we can transform G into H through an arbitrary sequence of the following two operations: (a) removal of an edge; (b) merge two connected vertices u, v: that is, remove edge (u, v) as well as vertices u and v; add a new vertex and make all edges incident on this new vertex that were incident on u or v. Now, if H is not a minor of G then we say that G excludes H as a minor.

The explanation of the following statement may help understand the definition better: *any graph H with r nodes is a minor of* K_r, where K_r is a complete graph of r nodes. This is true because one may obtain H by removing edges from K_r that are absent in H. More generally, if G is a subgraph of G' and G has H as a minor, then G' has H as its minor. Let $K_{r,r}$ denote a complete bipartite graph with r nodes in each partition. Then K_r is a minor of $K_{r,r}$. An important implication of this is as follows: to prove property P for graph G that excludes H, of size r, as a minor, it is sufficient to prove that any graph that excludes $K_{r,r}$ as a minor has property P. This fact was cleverly used by Klein et. al. [15].

B. Markov random field

A Markov Random Field (MRF) is defined on the basis of an undirected graph $G = (V, E)$ in the following manner. Let $V = \{1, \dots, n\}$ and $E \subset V \times V$. For each $v \in V$, let X_v be random variable taking values in some finite valued space Σ_v. Without loss of generality, lets assume that $\Sigma_v = \Sigma$ for all $v \in V$. Let $\mathbf{X} = (X_1, \dots, X_n)$ be the collection of these random variables taking values in Σ^n. For any subset $A \subset V$, we let $\mathbf{X}_A$ denote $\{X_v | v \in A\}$. We call a subset $S \subset V$ a *cut* of G if by its removal from G the graph decomposes into two or more disconnected components. That is, $V \backslash S = A \cup B$ with $A \cap B = \emptyset$ and there for any $a \in A, b \in B$, $(a, b) \notin E$. The $\mathbf{X}$ is called a Markov random field, if for any cut $S \subset V$, $\mathbf{X}_A$ and $\mathbf{X}_B$ are conditionally independent given $\mathbf{X}_S$, where $V \backslash S = A \cup B$.

By the Hammersley-Clifford theorem, any Markov random field that is strictly positive (i.e. $\Pr(\mathbf{X} = \mathbf{x}) > 0$ for all $\mathbf{x} \in \Sigma^n$) can be defined in terms of a decomposition of the distribution over cliques of the graph. Specifically, we will restrict our attention to pair-wise Markov random fields (to be defined precisely soon) only in this paper. This does not incur loss of generality for the following reason. A distributional representation that decomposes in terms of distribution over cliques can be represented through a factor graph over discrete variables. Any factor graph over discrete variables can be tranformed into a pair-wise Markov random field (see, [7] for example) by introducing auxiliary variables. As reader shall notice, the techniques of this paper can be extended to Markov random fields with higher-order interaction that contains hyper-edges.

Now, we present the precise definition of pair-wise Markov random field. We will consider distributions in exponential form. For each vertex $v \in V$ and edge $(u,v) \in E$, the corresponding potential functions are $\phi_v : \Sigma \to \mathbb{R}_+$ and $\psi_{uv} : \Sigma^2 \to \mathbb{R}_+$. Then, the distribution of $\mathbf{X}$ is given as follows: for $\mathbf{x} \in \Sigma^n$,

$$\Pr[\mathbf{X} = \mathbf{x}] \quad \propto \quad \exp\left(\sum_{v \in V} \phi_v(x_v) + \sum_{(u,v) \in E} \psi_{uv}(x_u, x_v)\right) (1)$$

We note that the assumption of ϕ_v, ψ_{uv} being non-negative does not incur loss of generality for the following reasons: (a) the distribution remains the same if we consider potential functions $\phi_v + C, \psi_{uv} + C$, for all $v \in V, (u,v) \in E$ with constant C; and (b) by selecting large enough constant, the modified functions will become non-negative as they are defined over finite discrete domain.

C. Log-partition function

The normalization constant in definition (1) of distribution is called the *partition function*, Z. Specifically,

$$Z = \sum_{\mathbf{x} \in \Sigma^n} \exp\left(\sum_{v \in V} \phi_v(x_v) + \sum_{(u,v) \in E} \psi_{uv}(x_u, x_v)\right).$$

Clearly, the knowledge of Z is necessary in order to evaluate probability distribution or to compute marginal probabilities, i.e. $\Pr(X_v = x_v)$ for $v \in V$. In applications in computer science, Z corresponds to the number of combinatorial objects, in statistical physics normalized logarithm of Z provides free-energy and in reversible stochastic networks Z provides loss probability for evaluating quality of service.

In this paper, we will be interested in obtaining estimate of $\log Z$. Specifically, we will call $\hat{Z}$ as an ε-approximation of Z if

$$(1-\varepsilon) \log Z \leq \log \hat{Z} \leq (1+\varepsilon) \log Z.$$

D. MAP assignment

The maximum a posteriori (MAP) assignment $\mathbf{x}^*$ is one with maximal probability, i.e.

$$\mathbf{x}^* \in \arg\max_{\mathbf{x} \in \Sigma^n} \Pr[\mathbf{X} = \mathbf{x}].$$

Computing MAP assignment is of interest in wide variety of applications. In combinatorial optimization problem, $\mathbf{x}^*$ corresponds to an optimizing solution, in the context of image processing it can be used as the basis for image segmentation techniques and in error-correcting codes it corresponds to decoding the received noisy code-word.

In our setup, MAP assignment $\mathbf{x}^*$ corresponds to

$$\mathbf{x}^* \in \arg\max_{\mathbf{x} \in \Sigma^n} \left(\sum_{v \in V} \phi_v(x_v) + \sum_{(u,v) \in E} \psi_{uv}(x_u, x_v)\right).$$

Define, $\mathcal{H}(\mathbf{x}) = (\sum_{v \in V} \phi_v(x_v) + \sum_{(u,v) \in E} \psi_{uv}(x_u, x_v))$. We will be interested in obtaining an ε estimate, say $\widehat{\mathbf{x}}$, of $\mathbf{x}^*$ such that

$$(1-\varepsilon)\mathcal{H}(\mathbf{x}^*) \leq \mathcal{H}(\widehat{\mathbf{x}}) \leq \mathcal{H}(\mathbf{x}^*).$$

III. Graph decomposition

In this section, we introduce notion of *graph decomposition*. We describe very simple algorithms for obtaining decomposition for graphs with low doubling dimension and minor-excluded graphs. In the later sections, we will show that such decomposable graphs are good structures in the sense that they allow for local algorithms for approximately computing log-partition function and MAP.

A. (ε, Δ) decomposition

Given $\varepsilon, \Delta > 0$, we define notion of (ε, Δ) decomposition for a graph $G = (V, E)$. This notion can be stated in terms of vertex-based decomposition or edge-based decomposition.

We call a random subset of vertices $\mathcal{B} \subset V$ as (ε, Δ) vertex-decomposition of G if the following holds: (a) For any $v \in V$, $\Pr(v \in \mathcal{B}) \leq \varepsilon$. (b) Let $S_1, \ldots, S_K$ be connected components of graph $G' = (V', E')$ where $V' = V \backslash \mathcal{B}$ and $E' = \{(u,v) \in E : u, v \in V'\}$. Then, $\max_{1 \leq k \leq K} |S_k| \leq \Delta$ with probability 1.

Similarly, a random subset of edges $\mathcal{B} \subset E$ is called an (ε, Δ) edge-decomposition of G if the following holds: (a) For any $e \in E$, $\Pr(e \in \mathcal{B}) \leq \varepsilon$. (b) Let $S_1, \ldots, S_K$ be connected components of graph $G' = (V', E')$ where $V' = V$ and $E' = E \backslash \mathcal{B}$. Then, $\max_{1 \leq k \leq K} |S_k| \leq \Delta$ with probability 1.

B. Low doubling-dimension graphs

This section presents (ε, Δ) decomposition algorithm for graphs with low doubling dimension for various choice of ε and Δ. Such a decomposition algorithm can be obtained through a probabilistically padded decomposition for such graphs [20]. However, we present our (different) algorithm due to its simplicity. Its worth noting that this simplicity of the algorithm requires proof technique different (and more complicated) than that known in the literature.

We will describe algorithm for node-based (ε, Δ) decomposition. This will immediately imply algorithm for edge-based decomposition for the following reason: given $G = (V, E)$ with doubling dimension $\rho(G)$, consider graph of its edges $\mathcal{G} = (E, \mathcal{E})$ where $(e, e') \in \mathcal{E}$ if e, e' shared a vertex in G. It is easy to check that $\rho(\mathcal{G}) \leq 2\rho(G) + 1$. Therefore, running algorithm for node-based decomposition on $\mathcal{G}$ will provide an edge-based decomposition.

The node-based decomposition algorithm for G will be described for the metric space on V with respect to the shortest path metric $\mathbf{d_G}$ introduced earlier. Clearly, it is not possible to have (ε, Δ) decomposition for any ε and Δ values. As will become clear later, it is important to have such decomposition for ε and Δ being not too large (specifically, we would like $\Delta = O(\log n)$). Therefore, we describe algorithm for any $\varepsilon > 0$ and an operational parameter K. We will show that the algorithm will produce (ε, Δ) node-decomposition where Δ will depend on ε, K and ρ.

Given ε and K, define random variable $\mathbf{Q}$ over $\{1, \ldots, K\}$ as

$$\Pr[\mathbf{Q} = i] = \begin{cases} \varepsilon(1-\varepsilon)^{i-1} & \text{if } 1 \leq i < K \\ (1-\varepsilon)^{K-1} & \text{if } i = K \end{cases}.$$

Define, $P_K = (1-\varepsilon)^{K-1}$. The algorithm DB-DIM$(\varepsilon, K)$ described next essentially does the following: initially, all vertices are colored *white*. Iteratively, choose a white vertex *arbitrarily*. Let u_t be vertex chosen in iteration t. Draw an independent random number as per distribution of $\mathbf{Q}$, say $\mathbf{Q}_t$. Select all *white* vertices that are at distance $\mathbf{Q}_t$ from u_t in $\mathcal{B}$ and color them *blue*; color all *white* vertices at distance $< \mathbf{Q}_t$ from u_t (including itself) as *red*. Repeat this process till no more *white* vertices are left. Output $\mathcal{B}$ (i.e. *blue* nodes) as the decomposition. Now, precise description of the algorithm.

```
DB-DIM$(\varepsilon, K)$

(1) Initially, set iteration number $t = 0$, $\mathcal{W}_0 = V$, $\mathcal{B}_0 = \emptyset$ and $\mathcal{R}_0 = \emptyset$.
(2) Repeat the following till $\mathcal{W}_t \neq \emptyset$:
    (a) Choose an element $u_t \in \mathcal{W}_t$ uniformly at random.
    (b) Draw a random number $Q_t$ independently according to the distribution of $\mathbf{Q}$.
    (c) Update
        (i) $\mathcal{B}_{t+1} \leftarrow \mathcal{B}_t \cup \{w | \mathbf{d_G}(u_t, w) = Q_t \text{ and } w \in \mathcal{W}_t\}$,
        (ii) $\mathcal{R}_{t+1} \leftarrow \mathcal{R}_t \cup \{w | \mathbf{d_G}(u_t, w) < Q_t \text{ and } w \in \mathcal{W}_t\}$,
        (iii) $\mathcal{W}_{t+1} \leftarrow \mathcal{W}_t \cap (\mathcal{B}_{t+1} \cup \mathcal{R}_{t+1})^c$.
    (d) Increment $t \leftarrow t + 1$.
(3) Output $\mathcal{B}_t$.
```

We state property of the algorithm DB-DIM(ε, K) as follows.

Lemma 2: Given G with doubling dimension $\rho = \rho(G)$ and $\varepsilon \in (0,1)$, let $K(\varepsilon, \rho) = \frac{12\rho}{\varepsilon} \log\left(\frac{24\rho}{\varepsilon}\right)$. Then DB-DIM$(\varepsilon, K(\varepsilon, \rho))$ produces random output $\mathcal{B} \subset V$ that is $(2\varepsilon, \Delta(\varepsilon, \rho))$ node-decomposition of G with $\Delta(\varepsilon, \rho) \leq K(\varepsilon, \rho)^{2\rho}$. The algorithm takes $O(C(\varepsilon, \rho)n)$ amount of time to produce $\mathcal{B}$, where $C(\varepsilon, \rho) = K(\varepsilon, \rho)^{2\rho}$.

Before presenting the proof of Lemma 2, we state the following important corollary for designing efficient algorithm.

Corollary 1: Let $\varepsilon \leq 1, \rho$ be such that $\rho \log(\rho/\varepsilon) = o(\log \log n)$. Then DB-DIM$(\varepsilon/2, K(\varepsilon/2, \rho))$ produces $(\varepsilon, \log^{1/L} n)$ node-decomposition for any finite (not scaling with n) L .

Proof: Since $\rho \log(\rho/\varepsilon) = o(\log \log n)$, we have that

$$2\rho\left(\log\frac{24\rho}{\varepsilon} + \log\log\frac{48\rho}{\varepsilon}\right) = o(\log\log n).$$

Therefore, by definition of $K(\varepsilon, \rho)$ we have that

$$\begin{aligned} K(\varepsilon/2, \rho)^{2\rho} &= \exp\left(2\rho\left[\log\frac{24\rho}{\varepsilon} + \log\log\frac{48\rho}{\varepsilon}\right]\right) \\ &= \exp\left(o(\log\log n)\right) \\ &\leq \log^{1/L} n, \end{aligned} \tag{2}$$

for any finite L. The last inequality follows from the definition of notation $o(\cdot)$. Now, Lemma 2 implies the desired claim. ∎

Proof: (Lemma 2) To prove claim of Lemma, we need to show two properties of the output set $\mathcal{B}$ for given ε and $K = K(\varepsilon, \rho)$: (a) for any $v \in V$, $\Pr(v \in \mathcal{B}) \leq 2\varepsilon$; (b) the graph G, upon removal of $\mathcal{B}$, decomposes into connected component each of size at most $K^{2\rho}$.

Before, we prove (a) and (b), lets bound the running time of the algorithm. Note that the algorithm runs for at most n iterations. In each iteration, the algorithm needs to check nodes that are within distance $K(\varepsilon, \rho)$ of the randomly chosen node. Therefore, total number of operations performed is at most $O(K(\varepsilon, \rho)^{2\rho})$. Thus, the total running time is $O(nK(\varepsilon, \rho)^{2\rho})$. Now we first justify (a) and then (b).

Proof of (a). To prove (a), we use the following Claim.

Claim 1: Consider metric space $\mathcal{M} = (V, \mathbf{d_G})$ with $|V| = n$. Let $\mathcal{B} \subset V$ be the random set that is output of decomposition algorithm with parameter (ε, K) applied to $\mathcal{M}$. Then, for any $v \in V$

$$\Pr[v \in \mathcal{B}] \leq \varepsilon + P_K |\mathbf{B}(v, K)|,$$

where $\mathbf{B}(v, K)$ is the ball of radius K in $\mathcal{M}$ with respect to the $\mathbf{d_G}$.

Proof: (Claim 1) The proof is by induction on the number of points n over which metric space is defined. When $n = 1$, the algorithm chooses only point as u_0 in the initial iteration and hence it can not be part of the output set $\mathcal{B}$. That is, for this only point, say v,

$$\Pr[v \in \mathcal{B}] = 0 \leq \varepsilon + P_K |\mathbf{B}(v, K)|.$$

Thus, we have verified the base case for induction ($n = 1$).

As induction hypothesis, suppose that the Claim 1 is true for any metric space on n points with $n < N$ for some $N \geq 2$. As the induction step, we wish to establish that for a metric space $\mathcal{M} = (V, \mathbf{d_G})$ with $|V| = N$, the Claim 1 is true. For this, consider any point $v \in V$. Now consider the first iteration of the algorithm applied to $\mathcal{M}$. The algorithm picks $u_0 \in V$ uniformly at random in the first iteration. Given v, depending on the choice of u_0 we consider four different cases (or events).

Case 1. This case corresponds to event E_1 where the chosen random u_0 is equal to point v of our interest. By definition of algorithm, under the event E_1, v will never be part of output set $\mathcal{B}$. That is,

$$\Pr[v \in \mathcal{B} | E_1] = 0 \;\leq\; \varepsilon + P_K |\mathbf{B}(v, K)|.$$

Case 2. Now, suppose u_0 be such that $v \neq u_0$ and $\mathbf{d_G}(u_0, v) < K$. Call this event E_2. Further, depending on choice of random number Q_0, define events:

$$E_{21} = \{\mathbf{d_G}(u_0, v) < Q_0\}, \; E_{22} = \{\mathbf{d_G}(u_0, v) = Q_0\}, \text{ and}$$
$$E_{23} = \{\mathbf{d_G}(u_0, v) > Q_0\}.$$

By definition of algorithm, when E_{21} happens, v is selected as part of $\mathcal{R}_1$ and hence can never be part of output $\mathcal{B}$. When E_{22} happens v is selected as part of $\mathcal{B}_1$ and hence it is definitely part of output set $\mathcal{B}$. When E_{23} happens, v is neither selected in set $\mathcal{R}_1$ nor selected in set $\mathcal{B}_1$. It is left as an element of the set $\mathcal{W}_1$. This new set $\mathcal{W}_1$ has points $< N$.

The original metric $\mathbf{d_G}$ is still a metric on points[1] of $\mathcal{W}_1$. By definition, the algorithm only cares about $(\mathcal{W}_1, \mathbf{d_G})$ in future and is not affected by its decisions in past. Therefore, we can invoke induction hypothesis which implies that if event E_{23} happens then the probability of $v \in \mathcal{B}$ is bounded above by $\varepsilon + P_K|\mathbf{B}(v,K)|$. Finally, let us relate the $\Pr[E_{21}|E_2]$ with $\Pr[E_{22}|E_2]$. Suppose $\mathbf{d_G}(u_0, v) = \ell < K$. By definition of probability distribution of $\mathbf{Q}$, we have

$$\Pr[E_{22}|E_2] \quad = \quad \varepsilon(1-\varepsilon)^{\ell-1}. \tag{3}$$

$$\begin{aligned}\Pr[E_{21}|E_2] \quad &= \quad (1-\varepsilon)^{K-1} + \sum_{j=\ell+1}^{K-1} \varepsilon(1-\varepsilon)^{j-1} \\ &= \quad (1-\varepsilon)^{\ell}. \end{aligned} \tag{4}$$

That is,

$$\Pr[E_{22}|E_2] = \frac{\varepsilon}{1-\varepsilon}\Pr[E_{21}|E_2].$$

Let $q \overset{\triangle}{=} \Pr[E_{21}|E_2]$. Then,

$$\begin{aligned}\Pr[v \in \mathcal{B}|E_2] \quad &= \quad \Pr[v \in \mathcal{B}|E_{21} \cap E_2]\Pr[E_{21}|E_2] \\ &\qquad + \Pr[v \in \mathcal{B}|E_{22} \cap E_2]\Pr[E_{22}|E_2] \\ &\qquad + \Pr[v \in \mathcal{B}|E_{23} \cap E_2]\Pr[E_{23}|E_2] \\ &= \quad \frac{\varepsilon q}{1-\varepsilon} + (\varepsilon + P_K|\mathbf{B}(v,K)|)\left(1 - \frac{q}{1-\varepsilon}\right) \\ &= \quad \varepsilon + P_K|\mathbf{B}(v,K)| - \frac{qP_K|\mathbf{B}(v,K)|}{1-\varepsilon} \\ &\le \quad \varepsilon + P_K|\mathbf{B}(v,K)|. \end{aligned} \tag{5}$$

Case 3. Now, suppose $u_0 \neq v$ is such that $\mathbf{d_G}(u_0, v) = K$. Call this event E_3. Further, let event $E_{31} = \{Q_0 = K\}$. Due to independence of selection of Q_0, $\Pr[E_{31}|E_3] = P_K$. Under event $E_{31} \cap E_3$, $v \in \mathcal{B}$ with probability 1. Therefore,

$$\begin{aligned}\Pr[v \in \mathcal{B}|E_3] \quad &= \quad \Pr[v \in \mathcal{B}|E_{31} \cap E_3]\Pr[E_{31}|E_3] \\ &\qquad + \Pr[v \in \mathcal{B}|E_{31}^c \cap E_3]\Pr[E_{31}^c|E_3] \\ &= \quad P_K + \Pr[v \in \mathcal{B}|E_{31}^c \cap E_3](1-P_K). \end{aligned} \tag{6}$$

Under event, $E_{31}^c \cap E_3$, we have $v \in \mathcal{W}_1$ and the remaining metric space $(\mathcal{W}_1, \mathbf{d_G})$. This metric space has $< N$ points. Further, the ball of radius K around v with respect to this new metric space has at most $|\mathbf{B}(v,K)| - 1$ points (this ball is with respect to the original metric space $\mathcal{M}$ of N points). We can invoke induction hypothesis for this new metric space (because of similar justification as in the previous case) to obtain

$$\Pr[v \in \mathcal{B}|E_{31}^c \cap E_3] \quad \le \quad \varepsilon + P_K(|\mathbf{B}(v,K)| - 1). \tag{7}$$

From (6) and (7), we have

$$\begin{aligned}\Pr[v \in \mathcal{B}|E_3] \quad &\le \quad P_K + (1-P_K)(\varepsilon + P_K(|\mathbf{B}(v,K)| - 1)) \\ &= \quad \varepsilon(1-P_K) + P_K|\mathbf{B}(v,K)| \\ &\qquad + P_K^2(1 - |\mathbf{B}(v,K)|) \\ &\le \quad \varepsilon + P_K|\mathbf{B}(v,K)|. \end{aligned} \tag{8}$$

Case 4. Finally, let E_4 be the event that $\mathbf{d_G}(u_0, v) > K$. Then, at the end of the first iteration of the algorithm, we again have the remaining metric space $(\mathcal{W}_1, \mathbf{d_G})$ such that $|\mathcal{W}_1| < N$. Hence, as before by induction hypothesis we will have

$$\Pr[v \in \mathcal{B}|E_4] \le \varepsilon + P_K|\mathbf{B}(v,K)|.$$

Now, the four cases are exhaustive and disjoint. That is, $\cup_{i=1}^4 E_i$ is the universe. Based on the above discussion, we obtain the following.

$$\begin{aligned}\Pr[v \in \mathcal{B}] \quad &= \quad \sum_{i=1}^4 \Pr[v \in \mathcal{B}|E_i]\Pr[E_i] \\ &\le \quad \left(\max_{i=1}^4 \Pr[v \in \mathcal{B}|E_i]\right)\left(\sum_{i=1}^4 \Pr[E_i]\right) \\ &\le \quad \varepsilon + P_K|\mathbf{B}(v,K)|. \end{aligned} \tag{9}$$

This completes the proof of Claim 1. ■

Now, we will use Claim 1 to complete the proof of (a). Lemma 1 for metric space with doubling dimension ρ and integer distances imply that,

$$|\mathbf{B}(v,K)| \le \left|\mathbf{B}\left(v, 2^{\lceil \log_2 K \rceil}\right)\right| \le 2^{\rho(\log_2 K + 1)} = (2K)^{\rho}.$$

Therefore, it is sufficient to show that

$$P_K(2K)^{\rho} \le \varepsilon.$$

Recall that $K(\varepsilon, \rho) = \frac{12\rho}{\varepsilon}\log\left(\frac{24\rho}{\varepsilon}\right)$, and $P_K = (1-\varepsilon)^{K-1}$. Hence,

$$\begin{aligned} K \quad &= \quad \frac{12\rho}{\varepsilon}\log\left(\frac{24\rho}{\varepsilon}\right) \\ &\ge \quad \frac{6\rho}{\varepsilon}\left(\log\left(\frac{24\rho}{\varepsilon}\right) + \log\log\left(\frac{24\rho}{\varepsilon}\right)\right) \\ &= \quad \frac{6\rho}{\varepsilon}\log 2K. \end{aligned} \tag{10}$$

Now since $K \ge 3$, we obtain that $K - 1 \ge \frac{4\rho}{\varepsilon}\log 2K$. Then, from $K \ge \frac{1}{\varepsilon}$ and $\rho \ge 1$,

$$K - 1 \ge \frac{2\rho}{\varepsilon}\log 2K + \frac{2}{\varepsilon}\log\frac{1}{\varepsilon}.$$

Note that $\log(1-\varepsilon)^{-1} \ge \log(1+\varepsilon) \ge \frac{\varepsilon}{2}$, for $\varepsilon \in (0,1)$. Hence,

$$(K-1)\log(1-\varepsilon)^{-1} \ge \rho\log 2K + \log\frac{1}{\varepsilon},$$

which implies

$$(1-\varepsilon)^{K-1}(2K)^{\rho} \le \varepsilon.$$

This completes the proof of (a) of Lemma 2.

Proof of (b). First we give some notations. Define $R_t = \mathcal{R}_t - \mathcal{R}_{t-1}$, $B_t = \mathcal{B}_t - \mathcal{B}_{t-1}$ and

$$\partial R_t = \{v \in V : v \notin R_t \text{ and } \exists\ v' \in R_t \text{ s.t. } \mathbf{d_G}(v, v') = 1\}.$$

The followings are straightforward observations implied by the algorithm: for any $t \ge 0$, (i) $R_t \cap \mathcal{R}_{t-1} = \emptyset$, (ii) $B_t \cap \mathcal{B}_{t-1} = \emptyset$, (iii) $R_t \subset \mathbf{B}(u_{t-1}, Q_{t-1})$, and (iv) $B_t \subset \mathbf{B}(u_{t-1}, Q_{t-1}+1) -$

[1] Note the following subtle but crucial point. We are not changing the metric $\mathbf{d_G}$ after we remove points from original set of points as part of the algorithm.

$\mathbf{B}(u_{t-1}, Q_{t-1})$. Now, we state a crucial claim for proving (b).

Claim 2: For all $t \geq 0$, $\partial R_t \subset \mathcal{B}_t$.

Proof: (Claim 2) We prove it by induction. Initially, $\partial R_0 = \mathcal{B}_0 = \emptyset$ and hence the claim is trivial. At the end of the first iteration, by definition of the algorithm

$$R_1 = \mathcal{R}_1 = \mathbf{B}(u_0, Q_0), \text{ and}$$
$$B_1 = \mathcal{B}_1 = \mathbf{B}(u_0, Q_0 + 1) - \mathbf{B}(u_0, Q_0).$$

Therefore, by definition $\partial R_1 = \mathcal{B}_1$. Thus, the base case of induction is verified. Now, as the hypothesis for induction suppose that $\partial R_t \subset \mathcal{B}_t$ for all $t \leq \ell$, for some $\ell \geq 1$. As induction step, we will establish that $\partial R_{\ell+1} \subset \mathcal{B}_{\ell+1}$.

Suppose to the contrary, that $\partial R_{\ell+1} \not\subset \mathcal{B}_{\ell+1}$. That is, there exists $v \in \partial R_{\ell+1}$ such that $v \notin \mathcal{B}_\ell$. By definition of algorithm, we have

$$R_{\ell+1} = \mathbf{B}(u_\ell, Q_\ell) - (\mathcal{R}_\ell \cup \mathcal{B}_\ell).$$

Therefore,

$$\partial R_{\ell+1} \subset (\mathbf{B}(u_\ell, Q_\ell + 1) - \mathbf{B}(u_\ell, Q_\ell)) \cup \mathcal{R}_\ell \cup \mathcal{B}_\ell.$$

Again, by definition of the algorithm we have

$$B_{\ell+1} = \mathbf{B}(u_\ell, Q_\ell + 1) - \mathbf{B}(u_\ell, Q_\ell) - \mathcal{R}_\ell - \mathcal{B}_\ell.$$

Therefore, $v \in B_{\ell+1}$ or $v \in \mathcal{R}_\ell \cup \mathcal{B}_\ell$. Recall that by definition of algorithm $\mathcal{B}_\ell \cap \mathcal{R}_\ell = \emptyset$. Since we have assumed that $v \notin \mathcal{B}_{\ell+1}$, it must be that $v \in \mathcal{R}_\ell$. That is, there exists $\ell' \leq \ell$ such that $v \in R_{\ell'}$. Now since $v \in \partial R_{\ell+1}$ by assumption, it must be that there exists $v' \in R_{\ell+1}$ such that $\mathbf{d_G}(v, v') = 1$. Since by definition $R_{\ell+1} \cap R_{\ell'} = \emptyset$, we have $v' \in \partial R_{\ell'}$. By induction hypothesis, this implies that $v' \in \mathcal{B}_{\ell'} \subset \mathcal{B}_\ell$. That is, $\mathcal{B}_\ell \cap R_{\ell+1} \neq \emptyset$, which is a contradiction to the definition of our algorithm. That is, our assumption that $\partial R_{\ell+1} \not\subset \mathcal{B}_{\ell+1}$ is false. Thus, we have established the inductive step. This completes the induction argument and proof of the Claim 2. ■

Now when the algorithm terminates (which must happen within n iterations), say the output set is $\mathcal{B}_T$ and $V - \mathcal{B}_T = \mathcal{R}_T$ for some T. As noted above, $\mathcal{R}_T$ is a union of disjoint sets $R_1, \dots, R_T$. We want to show that R_i, R_j are disconnected for any $1 \leq i < j \leq T$ using Claim 2. Suppose to the contrary that they are connected. That is, there exists $v \in R_i$ and $v' \in R_j$ such that $\mathbf{d_G}(v, v') = 1$. Since $R_i \cap R_j = \emptyset$, it must be that $v' \in \partial R_i, v \in \partial R_j$. From Claim 2 and fact that $\mathcal{B}_t \subset \mathcal{B}_{t+1}$ for all t, we have that $R_i \cap \mathcal{B} \neq \emptyset, R_j \cap \mathcal{B} \neq \emptyset$. This is contrary to the definition of the algorithm. Thus, we have established that $R_1, \dots, R_T$ are disconnected components whose union is $V - \mathcal{B}_T$. By definition, each of $R_i \subset \mathbf{B}(u_{i-1}, K)$. Thus, we have established that $V - \mathcal{B}_T$ is made of connected components, each of which is contained inside balls of radius K with respect to $\mathbf{d_G}$. Since, G has doubling dimension ρ, Lemma 1 implies that the size of any ball of radius K is at most $(2K)^\rho$. Given choice of $\varepsilon \leq 1$ and $\rho \geq 1$, we have that $K \geq 2$. Therefore, $(2K)^\rho \leq K^{2\rho}$. This completes the proof of (b) and that of Lemma 2. ■

C. Minor-excluded graphs

Here we describe a simple and explicit construction of decomposition for graphs that exclude certain finite sized graphs as their minor. This scheme is a direct adapation of a scheme proposed by Klein, Plotkin, Rao [15] and Rao [16]. We describe an (ε, Δ) node-decomposition scheme. Later, we describe how it can be modified to obtain (ε, Δ) edge-decomposition.

Suppose, we are given graph G that excludes graph $K_{r,r}$ as minor. Recall that if a graph excludes some graph G_r of r nodes as its minor then it excludes $K_{r,r}$ as its minor as well. In what follows and the rest of the paper, we will always assume r to be some finite number that does not scale with n (the number of nodes in G). The following algorithm for generating node-decomposition uses parameter Λ. Later we shall relate the parameter Λ to the decomposition property of the output.

```
MINOR-V$(G, r, \Lambda)$
(0) Input is graph $G = (V, E)$ and $r, \Lambda \in \mathbb{N}$. Initially, $i = 0$,
    $G_0 = G$, $\mathcal{B} = \emptyset$.
(1) For $i = 0, \dots, r-1$, do the following.
    (a) Let $S_1^i, \dots, S_{k_i}^i$ be the connected components of $G_i$.
    (b) For each $S_j^i, 1 \leq j \leq k_i$, pick an arbitrary node
        $v_j \in S_j^i$.
        ∘ Create a breadth-first search tree $\mathcal{T}_j^i$ rooted at $v_j$
          in $S_j^i$.
        ∘ Choose a number $L_j^i$ uniformly at random from
          $\{0, \dots, \Lambda - 1\}$.
        ∘ Let $\mathcal{B}_j^i$ be the set of nodes at level $L_j^i, \Lambda + L_j^i, 2\Lambda +$
          $L_j^i, \dots$ in $\mathcal{T}_j^i$.
        ∘ Update $\mathcal{B} = \mathcal{B} \cup_{j=1}^{k_i} \mathcal{B}_j^i$.
    (c) set $i = i + 1$.
(3) Output $\mathcal{B}$ and graph $G' = (V, E \backslash \mathcal{B})$.
```

As stated above, the basic idea is to use the following step recursively (upto depth r of recursion): in each connected component, say S, choose a node arbitrarily and create a breadth-first search tree, say $\mathcal{T}$. Choose a number, say L, uniformly at random from $\{0, \dots, \Lambda - 1\}$. Remove (and add to $\mathcal{B}$) all nodes that are at level $L + k\Lambda, k \geq 0$ in $\mathcal{T}$. Clearly, the total running time of such an algorithm is $O(r(n + |E|))$ for a graph $G = (V, E)$ with $|V| = n$; with possible parallel implementation across different connected components.

Figure 1 explains the algorithm for a line-graph of $n = 9$ nodes, which excludes $K_{2,2}$ as a minor. The example is about a sample run of MINOR-V$(G, 2, 3)$ (Figure 1 shows the first iteration of the algorithm).

The following is the result that was in essence proved in [15], [16].

Lemma 3: If G excludes $K_{r,r}$ as a minor. Let $\mathcal{B}$ be the output of MINOR-V(G, r, Λ). Then each connected component of $V \backslash \mathcal{B}$ has diameter of size $O(\Lambda)$.

Proof: This Lemma, for $r = 3$ was proved by Rao in [16] (Lemma 5 and Corollary 6 of [16]). The result is based on Theorem 4.2 of [15], which holds for any r. Therefore, the

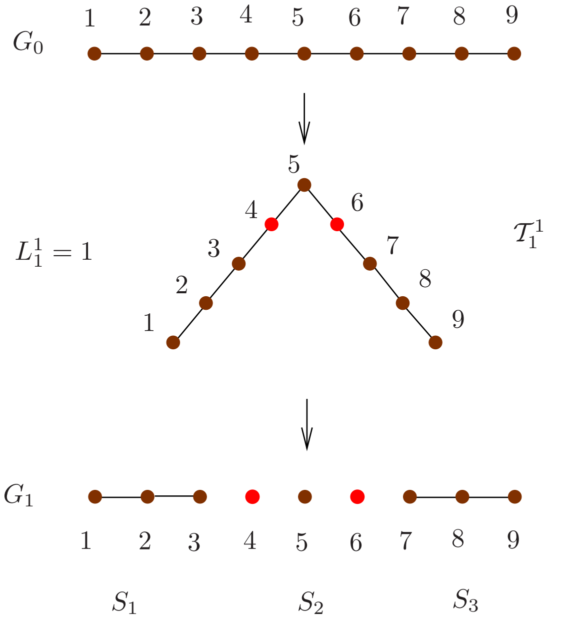

Fig. 1. The first of two iterations in execution of MINOR-V$(G, 2, 3)$ is shown.

result of Rao naturally extends for any r. This completes the justification of Lemma 3. ■

Now using Lemma 3, we obtain the following Lemma.

Lemma 4: Suppose G excludes $K_{r,r}$ as a minor. Let d^* be maximum vertex degree of nodes in G. Then algorithm MINOR-V(G, r, Λ) outputs $\mathcal{B}$ which is $(r/\Lambda, {d^*}^{O(\Lambda)})$ node-decomposition of G.

Proof: Let R be a connected component of $V\backslash\mathcal{B}$. From Lemma 3, the diameter of R is $O(\Lambda)$. Since d^* is the maximum vertex degree of nodes of G, the number of nodes in R is bounded above by ${d^*}^{O(\Lambda)}$.

To show that $\Pr(v \in \mathcal{B}) \le r/\Lambda$, consider a vertex $v \in V$. If $v \notin \mathcal{B}$ in the beginning of an iteration $0 \le i \le r-1$, then it will present in exactly one breadth-first search tree, say $\mathcal{T}^i_j$. This vertex v will be chosen in $\mathcal{B}^i_j$ only if it is at level $k\Lambda + L^i_j$ for some integer $k \ge 0$. The probability of this event is at most $1/\Lambda$ since L^i_j is chosen uniformly at random from $\{0, 1\dots, \Lambda-1\}$. By union bound, it follows that the probability that a vertex is chosen to be in $\mathcal{B}$ in any of the r iterations is at most r/Λ. This completes the proof of Lemma 4. ■

It is known that Planar graph excludes $K_{3,3}$ as a minor. Hence, Lemma 4 implies the following.

Corollary 2: Given a planar graph G with maximum vertex degree d^*, then the algorithm MINOR-V$(G, 3, \Lambda)$ produces $(3/\Lambda, {d^*}^{O(\Lambda)})$ node-decomposition for any $\Lambda \ge 1$.

We describe slight modification of MINOR-V to obtain algorithm that produces edge-decomposition as follows. Note that the only change compared to MINOR-V is the selection of edges rather than vertices to create the decomposition.

MINOR-E(G, r, Λ)

```
(0) Input is graph $G = (V, E)$ and $r, \Lambda \in \mathbb{N}$. Initially, $i = 0$,
    $G_0 = G$, $\mathcal{B} = \emptyset$.
(1) For $i = 0, \dots, r-1$, do the following.
    (a) Let $S^i_1, \dots, S^i_{k_i}$ be the connected components of $G_i$.
    (b) For each $S^i_j, 1 \le j \le k_i$, pick an arbitrary node
        $v_j \in S^i_j$.
        ○ Create a breadth-first search tree $\mathcal{T}^i_j$ rooted at $v_j$
          in $S^i_j$.
        ○ Choose a number $L^i_j$ uniformly at random from
          $\{0, \dots, \Lambda-1\}$.
        ○ Let $\mathcal{B}^i_j$ be the set of edges at level $L^i_j, \Lambda+L^i_j, 2\Lambda+$
          $L^i_j, \dots$ in $\mathcal{T}^i_j$.
        ○ Update $\mathcal{B} = \mathcal{B} \cup_{j=1}^{k_i} \mathcal{B}^i_j$.
    (c) set $i = i+1$.
(3) Output $\mathcal{B}$ and graph $G' = (V, E\backslash\mathcal{B})$.
```

Lemma 5: Suppose G excludes $K_{r,r}$ as a minor. Let d^* be maximum vertex degree of nodes in G. Then algorithm MINOR-E(G, r, Λ) outputs $\mathcal{B}$ which is $(r/\Lambda, {d^*}^{O(\Lambda)})$ edge-decomposition of G.

Proof: Let G^* be a graph that is obtained from G by adding center vertex to each edge of G. It is easy to see that if G excludes $K_{r,r}$ as minor then so does G^*.

Now the algorithm MINOR-E(G, r, Λ) can be viewed as executing MINOR-V$(G^*, r, 2\Lambda$-1) with modification that the random numbers L^i_js are chosen uniformly at random from $\{1, 3, 5, \dots 2\Lambda-1\}$ instead of the whole support $\{1, 2, \dots, 2\Lambda-1\}$. To prove Lemma 5, we need to show that: (a) each edge is part of the output set $\mathcal{B}$ with probability at most r/Λ, and (b) each of the connected component of $V\backslash\mathcal{B}$ is at most ${d^*}^{O(\Lambda)}$.

The (a) follows from exactly the same arguments as those used in Lemma 4. For (b), consider the following. The Lemma 3 implies that if the algorithm was executed with the random numbers L^i_js being chosen from $\{1, 2, \dots, 2\Lambda-1\}$, then the desired result follows with probability 1. It is easy to see that under the execution of the algorithm with these choices for random numbers, with strictly positive probability (independent of n) all the L^i_js are chosen only from the odd numbers, i.e. $\{1, 3, 5, \dots 2\Lambda-1\}$. Therefore, it must be that when we restrict the choice of numbers to these odd numbers, the algorithm must produce the desired result. This completes the proof of Lemma 5. ■

Figure 2 explains the algorithm for a line-graph of $n = 9$ nodes, which excludes $K_{2,2}$ as a minor. The example is about a sample run of MINOR-E$(G, 2, 3)$ (Figure 2 shows the first iteration of the algorithm).

IV. APPROXIMATE $\log Z$

Here, we describe algorithm for approximate computation of $\log Z$ for any graph G. The algorithm uses an edge-decomposition algorithm as a sub-routine. Our algorithm provides provable upper and lower bound on $\log Z$ for any graph G. In order to obtain tight approximation guarantee, we will use specific graph structures as in low doubling dimension and minor-excluded graph.

A. Algorithm

In what follows, we use term DECOMP for a generic edge-decomposition algorithm. The approximation guarantee of the

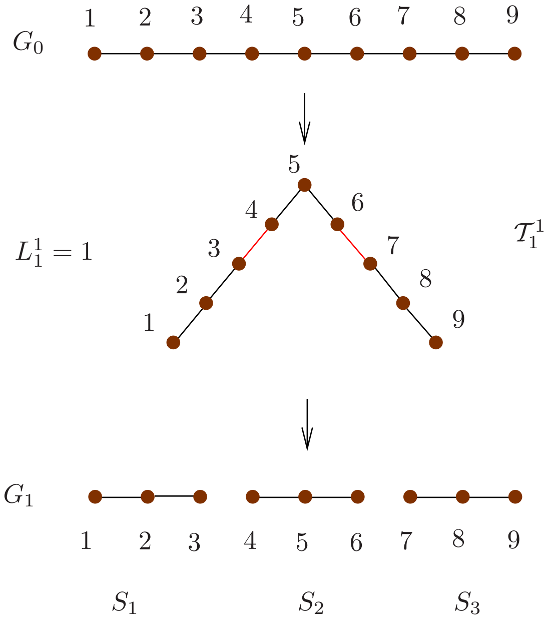

Fig. 2. The first of two iterations in execution of MINOR-E$(G, 2, 3)$ is shown.

output of the algorithm and its computation time depend on the property of DECOMP. For graph with low doubling dimension, we use algorithm DB-DIM(over the edge graph) and for graph that excludes $K_{r,r}$ as minor for some r, we use algorithm MINOR-E.

```
LOG PARTITION($G$)
(1) Use DECOMP($G$) to obtain $\mathcal{B} \subset E$ such that
    (a) $G' = (V, E \backslash \mathcal{B})$ is made of connected components $S_1, \dots, S_K$.
(2) For each connected component $S_j, 1 \le j \le K$, do the following:
    (a) Compute partition function $Z_j$ restricted to $S_j$ by dynamic programming (or exhaustive computation).
(3) Let $\psi_{ij}^L = \min_{(x,x') \in \Sigma^2} \psi_{ij}(x, x')$, $\psi_{ij}^U = \max_{(x,x') \in \Sigma^2} \psi_{ij}(x, x')$. Then
        $\log \hat{Z}_{\text{LB}} = \sum_{j=1}^{K} \log Z_j + \sum_{(i,j) \in \mathcal{B}} \psi_{ij}^L;$
        $\log \hat{Z}_{\text{UB}} = \sum_{j=1}^{K} \log Z_j + \sum_{(i,j) \in \mathcal{B}} \psi_{ij}^U.$
(4) Output: lower bound $\log \hat{Z}_{\text{LB}}$ and upper bound $\log \hat{Z}_{\text{UB}}$.
```

In words, LOG PARTITION(G) produces upper and lower bound on $\log Z$ of MRF G as follows: decompose graph G into (small) components $S_1, \dots, S_K$ by removing (few) edges $\mathcal{B} \subset E$ using DECOMP(G). Compute exact log-partition function in each of the components. To produce bounds $\log \hat{Z}_{\text{LB}}, \log \hat{Z}_{\text{UB}}$ take the summation of thus computed component-wise log-partition function along with minimal and maximal effect of edges from $\mathcal{B}$.

B. Analysis of LOG PARTITION*: General G*

Here, we analyze performance of LOG PARTITION for any G. Later, we will use property of the specific graph structure to obtain sharper approximation guarantees.

Theorem 1: Given a pair-wise MRF G, the LOG PARTITION produces $\log \hat{Z}_{\text{LB}}, \log \hat{Z}_{\text{UB}}$ such that

$$\log \hat{Z}_{\text{LB}} \le \log Z \le \log \hat{Z}_{\text{UB}},$$
$$\log \hat{Z}_{\text{UB}} - \log \hat{Z}_{\text{LB}} = \sum_{(i,j) \in \mathcal{B}} \left(\psi_{ij}^U - \psi_{ij}^L \right).$$

It takes $O\left(|E||\Sigma|^{|S^*|} \right) + T_{\text{DECOMP}}$ time to produce this estimate, where $|S^*| = \max_{j=1}^{K} |S_j|$ with DECOMP producing decomposition of G into $S_1, \dots, S_K$ in time T_{DECOMP}.

Proof: First, we prove properties of $\log \hat{Z}_{\text{LB}}, \log \hat{Z}_{\text{UB}}$ as follows:

$$\begin{aligned}
\log \hat{Z}_{\text{LB}} &\overset{(o)}{=} \sum_{j=1}^{K} \log Z_j + \sum_{(i,j) \in \mathcal{B}} \psi_{ij}^L \\
&\overset{(a)}{=} \log \Bigg[\sum_{\mathbf{x} \in \Sigma^n} \exp \Bigg(\sum_{i \in V} \phi_i(x_i) \\
&\qquad + \sum_{(i,j) \in E \backslash \mathcal{B}} \psi_{ij}(x_i, x_j) + \sum_{(i,j) \in \mathcal{B}} \psi_{ij}^L \Bigg) \Bigg] \\
&\overset{(b)}{\le} \log \Bigg[\sum_{\mathbf{x} \in \Sigma^n} \exp \Bigg(\sum_{i \in V} \phi_i(x_i) \\
&\qquad + \sum_{(i,j) \in E \backslash \mathcal{B}} \psi_{ij}(x_i, x_j) + \sum_{(i,j) \in \mathcal{B}} \psi_{ij}(x_i, x_j) \Bigg) \Bigg] \\
&= \log Z \\
&\overset{(c)}{\le} \log \Bigg[\sum_{\mathbf{x} \in \Sigma^n} \exp \Bigg(\sum_{i \in V} \phi_i(x_i) \\
&\qquad + \sum_{(i,j) \in E \backslash \mathcal{B}} \psi_{ij}(x_i, x_j) + \sum_{(i,j) \in \mathcal{B}} \psi_{ij}^U \Bigg) \Bigg] \\
&\overset{(d)}{=} \sum_{j=1}^{K} \log Z_j + \sum_{(i,j) \in \mathcal{B}} \psi_{ij}^U \\
&= \log \hat{Z}_{\text{UB}}.
\end{aligned}$$

We justify (a)-(d) as follows: (a) holds because by removal of edges $\mathcal{B}$, the G decomposes into disjoint connected components $S_1, \dots, S_K$; (b) holds because of the definition of ψ_{ij}^L; (c) holds by definition ψ_{ij}^U and (d) holds for a similar reason as (a). The claim about difference $\log \hat{Z}_{\text{UB}} - \log \hat{Z}_{\text{LB}}$ in the statement of Theorem 1 follows directly from definitions (i.e. subtract RHS (o) from (d)). This completes proof of claimed relation between bounds $\log \hat{Z}_{\text{LB}}, \log \hat{Z}_{\text{UB}}$.

For running time analysis, note that LOG PARTITION performs two main tasks: (i) Decomposing G using DECOMP algorithm, which by definition take T_{DECOMP} time. (ii) Computing Z_j for each component S_j through exhaustive computation, which takes $O(|E_j||\Sigma|^{|S_j|})$ time (where E_j are edges between nodes of S_j) and producing $\log \hat{Z}_{\text{LB}}, \log \hat{Z}_{\text{UB}}$ takes addition $|E|$ operations at the most. Now, the maximum size among these components is $|S^*|$. Further, the $\cup_j E_j \subset E$. Therefore, we obtain that the total running time for this task is $O(|E||\Sigma|^{|S^*|})$. Putting (i) and (ii) together, we obtain the desired bound. This completes the proof of Theorem 1. ■

C. *Some preliminaries*

Before stating precise approximation bound of LOG PARTITION algorithm for graphs with low doubling dimension and graphs that exclude minors, we state two useful Lemmas about $\log Z$ for any graph.

Lemma 6: If G has maximum vertex degree d^* then,

$$\log Z \geq \frac{1}{d^*+1}\left[\sum_{(i,j)\in E} \psi_{ij}^U - \psi_{ij}^L\right].$$

Proof: Assign weight $w_{ij} = \psi_{ij}^U - \psi_{ij}^L$ to an edge $(i,j) \in E$. Since graph has maximum vertex degree d^*, by Vizing's theorem there exists an edge-coloring of the graph using at most d^*+1 colors. Edges with the same color form a matching of the G. A standard application of Pigeon-hole's principle implies that there is a color with weight at least $\frac{1}{d^*+1}(\sum_{(i,j)\in E} w_{ij})$. Let $M \subset E$ denote these set of edges. That is,

$$\sum_{(i,j)\in M} (\psi_{ij}^U - \psi_{ij}^L) \geq \frac{1}{d^*+1}\left(\sum_{(i,j)\in E} (\psi_{ij}^U - \psi_{ij}^L)\right).$$

Now, consider a $Q \subset \Sigma^n$ of size $2^{|M|}$ created as follows. For $(i,j) \in M$ let $(x_i^U, x_j^U) \in \arg\max_{(x,x')\in\Sigma^2} \psi_{ij}(x,x')$. For each $i \in V$, choose $x_i^L \in \Sigma$ arbitrarily. Then,

$$\begin{aligned} Q &= \{\mathbf{x} \in \Sigma^n : \forall\ (i,j) \in M,\ (x_i,x_j) = (x_i^U, x_j^U) \text{ or} \\ &\quad (x_i^L, x_j^L); \text{for all other } i \in V,\ x_i = x_i^L\}. \end{aligned}$$

Note that we have used the fact that M is a matching for Q to be well-defined.

By definition ϕ_i, ψ_{ij} are non-negative function (hence, their exponents are at least 1). Using this property, we have the following:

$$\begin{aligned} Z &\geq \left[\sum_{\mathbf{x}\in Q} \exp\left(\sum_{i\in V}\phi_i(x_i) + \sum_{(i,j)\in E}\psi_{ij}(x_i,x_j)\right)\right] \\ &\overset{(o)}{\geq} \left[\sum_{\mathbf{x}\in Q} \exp\left(\sum_{(i,j)\in M}\psi_{ij}(x_i,x_j)\right)\right] \\ &\overset{(a)}{\geq} 2^{|M|}\prod_{(i,j)\in M}\frac{\exp(\psi_{ij}^L) + \exp(\psi_{ij}^U)}{2} \\ &\overset{(b)}{=} \prod_{(i,j)\in M}(1+\exp(\psi_{ij}^U - \psi_{ij}^L))\exp(\psi_{ij}^L) \\ &\overset{(c)}{\geq} \prod_{(i,j)\in M}\exp(\psi_{ij}^U - \psi_{ij}^L) \\ &= \exp\left(\sum_{(i,j)\in M}\psi_{ij}^U - \psi_{ij}^L\right). \end{aligned} \tag{11}$$

Justification of (o)-(c): (o) follows since ψ_{ij}, ϕ_i are non-negative functions. (a) consider the following probabilistic experiment: assign (x_i, x_j) for each $(i,j) \in M$ equal to (x_i^U, x_j^U) or (x_i^L, x_j^L) with probability $1/2$ each. Under this experiment, the expected value of the $\exp(\sum_{(i,j)\in M}\psi_{ij}(x_i,x_j))$, which is $\prod_{(i,j)\in M}\frac{\exp(\psi_{ij}(x_i^L,x_j^L))+\exp(\psi_{ij}(x_i^U,x_j^U))}{2}$, is equal to $2^{-|M|}[\sum_{\mathbf{x}\in Q}\exp(\sum_{(i,j)\in M}\psi_{ij}(x_i,x_j))]$. Now, use the fact that $\psi_{ij}(x_i^L, x_j^L) \geq \psi_{ij}^L$. (b) follows from simple algebra and (c) follows by using non-negativity of function ψ_{ij}. Therefore,

$$\begin{aligned} \log Z &\geq \sum_{(i,j)\in M}(\psi_{ij}^U - \psi_{ij}^L) \\ &\geq \frac{1}{d^*+1}\left(\sum_{(i,j)\in E}(\psi_{ij}^U - \psi_{ij}^L)\right), \end{aligned} \tag{12}$$

using fact about weight of M. This completes the proof of Lemma 6. ■

Lemma 7: If G has maximum vertex degree d^* and the DECOMP(G) produces $\mathcal{B}$ that is (ε, Δ) edge-decomposition, then

$$\mathbb{E}\left[\log\hat{Z}_{\text{UB}} - \log\hat{Z}_{\text{LB}}\right] \leq \varepsilon(d^*+1)\log Z,$$

w.r.t. the randomness in $\mathcal{B}$, and LOG PARTITION takes time $O(nd^*|\Sigma|^{\Delta}) + T_{\text{DECOMP}}$.

Proof: From Theorem 1, Lemma 6 and definition of (ε, Δ) edge-decomposition, we have the following.

$$\begin{aligned} \mathbb{E}\left[\log\hat{Z}_{\text{UB}} - \log\hat{Z}_{\text{LB}}\right] &\leq \mathbb{E}\left[\sum_{(i,j)\in\mathcal{B}}(\psi_{ij}^U - \psi_{ij}^L)\right] \\ &= \sum_{(i,j)\in E}\Pr((i,j)\in\mathcal{B})(\psi_{ij}^U - \psi_{ij}^L) \\ &\leq \varepsilon\left[\sum_{(i,j)\in E}(\psi_{ij}^U - \psi_{ij}^L)\right] \\ &\leq \varepsilon(d^*+1)\log Z. \end{aligned}$$

Now to estimate the running time, note that under (ε, Δ) decomposition $\mathcal{B}$, with probability 1 the $G' = (V, E\backslash\mathcal{B})$ is divided into connected components with at most Δ nodes. Therefore, the running time bound of Theorem 1 implies the desired result. ■

D. *Analysis of* LOG PARTITION*: Low doubling-dimension G*

Here we interpret result obtained in Theorem 1 and Lemma 7, for G that has low doubling-dimension and uses decomposition scheme DB-DIM.

Theorem 2: Let MRF graph G of n nodes with doubling dimension ρ be given. Consider any $\varepsilon \in (0,1)$. Define $\varphi = \varepsilon 2^{-\rho-3}$. Then Log Partition using Db-dim$(\varphi, K(\varphi,\rho))$ produces bounds $\log \hat{Z}_{\text{LB}}, \log \hat{Z}_{\text{UB}}$ such that

$$\mathbb{E}\left[\log \hat{Z}_{\text{UB}} - \log \hat{Z}_{\text{LB}}\right] \le \varepsilon \log Z.$$

The algorithm takes $O(n2^{\rho} C_0(\varepsilon,\rho))$ time to obtain the estimate, where $C_0(\varepsilon,\rho) = |\Sigma|^{K(\varphi,\rho)^{2\rho}}$. Further, if $\rho(\rho + \log 1/\varepsilon) = o(\log\log n)$ then the algorithm takes $o(n^{1+\delta})$ amount of time for any $\delta > 0$.

Proof: The Lemma 2, Lemma 7 and Theorem 1 implies the following bound:

$$\begin{aligned}\mathbb{E}\left[\log \hat{Z}_{\text{UB}} - \log \hat{Z}_{\text{LB}}\right] &\le \varepsilon 2^{-\rho-2}(d^*+1)\log Z \\ &\le \varepsilon \log Z. \qquad (13)\end{aligned}$$

Now for graph with doubling dimension ρ, $|E| = n2^{\rho}$. Under the decomposition algorithm with parameter φ and $K(\varphi,\rho)$, the number of nodes in any component is at most $K(\varphi,\rho)^{2\rho}$. Therefore, by Lemma 2 the desired bound on running time follows.

Now, consider when condition $\rho(\rho + \log 1/\varepsilon) = o(\log\log n)$. Given $\varphi = \varepsilon 2^{-\rho-3}$,

$$\begin{aligned}\rho \log \rho/\varphi &= \rho(\log\rho + \rho + 3 + \log 1/\varepsilon) \\ &= \Theta(\rho^2 + \rho\log 1/\varepsilon) \\ &= o(\log\log n), \qquad (14)\end{aligned}$$

from the above described hypothesis of the Theorem. Now, Db-dim$(\varphi, K(\varphi,\rho))$ produces $(\varepsilon 2^{-\rho-2}, O(\log^{1/L} n))$ edge-decomposition from Corollary 1. We select $L = 2$. Given this and above arguments, we have that the running time of the algorithm is $o(n^{1+\delta}))$ for any $\delta > 0$. This completes the proof of Theorem 2. ■

E. Analysis of Log Partition*: Minor-excluded G*

We apply Theorem 1 and Lemma 7 for minor-excluded graphs when the Decomp procedure is essentially the Minor-e. We obtain the following precise result.

Theorem 3: Let MRF graph G of n nodes exclude $K_{r,r}$ as its minor. Let d^* be the maximum vertex degree in G. Given $\varepsilon > 0$, use Log Partition algorithm with Minor-e(G, r, Λ) where $\Lambda = \lceil \frac{r(d^*+1)}{\varepsilon} \rceil$. Then,

$$\log \hat{Z}_{\text{LB}} \le \log Z \le \log \hat{Z}_{\text{UB}}; \text{ and}$$
$$\mathbb{E}\left[\log \hat{Z}_{\text{UB}} - \log \hat{Z}_{\text{LB}}\right] \le \varepsilon \log Z.$$

Further, algorithm takes $(nC(d^*,|\Sigma|,\varepsilon))$, where constant $C(d^*,|\Sigma|,\varepsilon) = d^*|\Sigma|^{d^{*O(\Lambda)}}$. Therefore, if $\varepsilon^{-1}d^*\log d^* = o(\log\log n)$, then the algorithm takes $o(n^{1+\delta})$ steps for arbitrary $\delta > 0$.

Proof: From Lemma 5 about the Minor-e algorithm, we have that with choice of $\Lambda = \lceil \frac{r(d^*+1)}{\varepsilon} \rceil$, the algorithm produces (ε, Δ) edge-decomposition where $\Delta = d^{*O(\Lambda)}$. Since it is an (ε, Δ) edge-decomposition, the upper bound and the lower bound, $\log \hat{Z}_{\text{UB}}, \log \hat{Z}_{\text{LB}}$, for the value produced by the algorithm are within $(1 \pm \varepsilon)\log Z$ by Lemma 7.

Now, by Lemma 7 the running time of the algorithm is $O(nd^*|\Sigma|^{\Delta}) + T_{\text{DECOMP}}$. As discussed earlier in Lemma 5, the algorithm Minor-e takes $O(r|E|) = O(nrd^*)$ operations. That is, $T_{\text{DECOMP}} = O(nrd^*)$. Now, $\Delta = d^{*O(\Lambda)}$ and $\Lambda \le r(d^*+1)/\varepsilon + 1$. Therefore, the first term of the computation time bound is bounded above by

$$O\left(nd^*|\Sigma|^{d^{*O(rd^*/\varepsilon)}}\right).$$

Now, we will establish that the above term is $O(n^2)$ under the hypothesis $\varepsilon^{-1}d^*\log d^* = o(\log\log n)$. The hypothesis implies that (since r a constant, not scaling with n):

$$\Lambda \log d^* = o(\log\log n).$$

That is, for any finite L (say, $L = 2$) we have that

$$\Delta = O(\log^{1/L} n).$$

This in turn implies that, for finite $|\Sigma|$ we have

$$|\Sigma|^{\Delta} = o(n^{\delta/2}),$$

for any $\delta > 0$. Since $d^* = o(\log\log n) = O(n^{\delta/2})$. Therefore, it follows that

$$O\left(nd^*|\Sigma|^{\Delta}\right) = o(n^{1+\delta}).$$

This completes the proof of Theorem 3. ■

V. Approximate MAP

Now, we describe algorithm to compute MAP approximately. It is very similar to the Log Partition algorithm: given G, decompose it into (small) components $S_1, \dots, S_K$ by removing (few) edges $\mathcal{B} \subset E$. Then, compute an approximate MAP assignment by computing exact MAP restricted to the components. As in Log Partition, the computation time and performance of the algorithm depends on property of decomposition scheme. We describe algorithm for any graph G; which will be specialized for graph with low doubling dimension and graph that exclude minor by using the appropriate edge-decomposition schemes.

```
Mode(G)

(0) Input is MRF $G = (V,E)$ with $\phi_i(\cdot), i \in V$,
    $\psi_{ij}(\cdot,\cdot), (i,j) \in E$.
(1) Use Decomp(G) to obtain $\mathcal{B} \subset E$ such that
    (a) $G' = (V, E \backslash \mathcal{B})$ is made of connected components
        $S_1, \dots, S_K$.
(2) For each connected component $S_j, 1 \le j \le K$, do the
    following:
    (a) Through dynamic programming (or exhaustive com-
        putation) find exact MAP $\mathbf{x}^{*,j}$ for component $S_j$,
        where $\mathbf{x}^{*,j} = (x_i^{*,j})_{i \in S_j}$.
(3) Produce output $\widehat{\mathbf{x}^*}$, which is obtained by assigning values
    to nodes using $\mathbf{x}^{*,j}, 1 \le j \le K$.
```

A. Analysis of MODE: General G

Here, we analyze performance of MODE for any G. Later, we will specialize our analysis for graph with low doubling dimension and minor excluded graphs.

Theorem 4: Given an MRF G described by (1), the MODE algorithm produces outputs $\widehat{\mathbf{x}^*}$ such that:

$$\mathcal{H}(\mathbf{x}^*) - \sum_{(i,j)\in\mathcal{B}} \left(\psi_{ij}^U - \psi_{ij}^L\right) \le \mathcal{H}(\widehat{\mathbf{x}^*}) \le \mathcal{H}(\mathbf{x}^*).$$

The algorithm takes $O\left(|E|K|\Sigma|^{|S^*|}\right) + T_{\text{DECOMP}}$ time to produce this estimate, where $|S^*| = \max_{j=1}^K |S_j|$ with DECOMP producing decomposition of G into $S_1, \dots, S_K$ in time T_{DECOMP}.

Proof: By definition of MAP $\mathbf{x}^*$, we have $\mathcal{H}(\widehat{\mathbf{x}^*}) \le \mathcal{H}(\mathbf{x}^*)$. Now, consider the following.

$$\begin{aligned}
\mathcal{H}(\mathbf{x}^*) &= \max_{\mathbf{x}\in\Sigma^n}\left[\sum_{i\in V}\phi_i(x_i) + \sum_{(i,j)\in E}\psi_{ij}(x_i,x_j)\right] \\
&= \max_{\mathbf{x}\in\Sigma^n}\left[\sum_{i\in V}\phi_i(x_i) + \sum_{(i,j)\in E\backslash\mathcal{B}}\psi_{ij}(x_i,x_j)\right. \\
&\qquad \left. + \sum_{(i,j)\in\mathcal{B}}\psi_{ij}(x_i,x_j)\right] \\
&\overset{(a)}{\le} \max_{\mathbf{x}\in\Sigma^n}\left[\sum_{i\in V}\phi_i(x_i) + \sum_{(i,j)\in E\backslash\mathcal{B}}\psi_{ij}(x_i,x_j)\right. \\
&\qquad \left. + \sum_{(i,j)\in\mathcal{B}}\psi_{ij}^U\right] \\
&\overset{(b)}{=} \sum_{j=1}^K\left[\max_{\mathbf{x}^j\in\Sigma^{|S_j|}}\mathcal{H}(\mathbf{x}^j)\right] + \left[\sum_{(i,j)\in\mathcal{B}}\psi_{ij}^U\right] \\
&\overset{(c)}{=} \sum_{j=1}^K\mathcal{H}(\mathbf{x}^{*,j}) + \left[\sum_{(i,j)\in\mathcal{B}}\psi_{ij}^U\right] \\
&\overset{(d)}{\le} \mathcal{H}(\widehat{\mathbf{x}^*}) + \left[\sum_{(i,j)\in\mathcal{B}}\psi_{ij}^U - \psi_{ij}^L\right]. \qquad (15)
\end{aligned}$$

We justify (a)-(d) as follows: (a) holds because for each edge $(i,j)\in\mathcal{B}$, we have replaced its effect by maximal value ψ_{ij}^U; (b) holds because by placing constant value ψ_{ij}^U over $(i,j)\in\mathcal{B}$, the maximization over G decomposes into maximization over the connected components of $G' = (V, E\backslash\mathcal{B})$; (c) holds by definition of $\mathbf{x}^{*,j}$ and (d) holds because when we obtain global assignment $\widehat{\mathbf{x}^*}$ from $\mathbf{x}^{*,j}, 1\le j\le K$ and compute its global value, the additional terms get added for each $(i,j)\in\mathcal{B}$ which add at least ψ_{ij}^L amount.

The running time analysis of MODE is exactly the same as that of LOG PARTITION in Theorem 1. Hence, we skip the details here. This completes the proof of Theorem 4. ■

B. Some preliminaries

This section presents some results about the property of MAP solution that will be useful in obtaining tight approximation guarantees later. First, consider the following.

Lemma 8: If G has maximum vertex degree d^*, then

$$\begin{aligned}
\mathcal{H}(\mathbf{x}^*) &\ge \frac{1}{d^*+1}\left[\sum_{(i,j)\in E}\psi_{ij}^U\right] \\
&\ge \frac{1}{d^*+1}\left[\sum_{(i,j)\in E}\psi_{ij}^U - \psi_{ij}^L\right]. \qquad (16)
\end{aligned}$$

Proof: Assign weight $w_{ij} = \psi_{ij}^U$ to an edge $(i,j)\in E$. Using argument of Lemma 6, we obtain that there exists a matching $M\subset E$ such that

$$\sum_{(i,j)\in M}\psi_{ij}^U \ge \frac{1}{d^*+1}\left(\sum_{(i,j)\in E}\psi_{ij}^U\right).$$

Now, consider an assignment $\mathbf{x}^M$ as follows: for each $(i,j)\in M$ set $(x_i^M, x_j^M) = \arg\max_{(x,x')\in\Sigma^2}\psi_{ij}(x,x')$; for remaining $i\in V$, set x_i^M to some value in Σ arbitrarily. Note that for above assignment to be possible, we have used matching property of M. Therefore, we have

$$\begin{aligned}
\mathcal{H}(\mathbf{x}^M) &= \sum_{i\in V}\phi_i(x_i^M) + \sum_{(i,j)\in E}\psi_{ij}(x_i^M, x_j^M) \\
&= \sum_{i\in V}\phi_i(x_i^M) + \sum_{(i,j)\in E\backslash M}\psi_{ij}(x_i^M, x_j^M) \\
&\qquad + \sum_{(i,j)\in M}\psi_{ij}(x_i^M, x_j^M) \\
&\overset{(a)}{\ge} \sum_{(i,j)\in M}\psi_{ij}(x_i^M, x_j^M) \\
&= \sum_{(i,j)\in M}\psi_{ij}^U \\
&\ge \frac{1}{d^*+1}\left[\sum_{(i,j)\in E}\psi_{ij}^U\right]. \qquad (17)
\end{aligned}$$

Here (a) follows because ψ_{ij}, ϕ_i are non-negative valued functions. Since $\mathcal{H}(\mathbf{x}^*) \ge \mathcal{H}(\mathbf{x}^M)$ and $\psi_{ij}^L \ge 0$ for all $(i,j)\in E$, we obtain the Lemma 8. ■

Lemma 9: If G has maximum vertex degree d^* and the DECOMP(G) produces $\mathcal{B}$ that is (ε,Δ) edge-decomposition, then

$$\mathbb{E}\left[\mathcal{H}(\mathbf{x}^*) - \mathcal{H}(\widehat{\mathbf{x}^*})\right] \le \varepsilon(d^*+1)\mathcal{H}(\mathbf{x}^*),$$

where expectation is w.r.t. the randomness in $\mathcal{B}$. Further, MODE takes time $O(nd^*|\Sigma|^\Delta) + T_{\text{DECOMP}}$.

Proof: From Theorem 4, Lemma 8 and definition of (ε, Δ) edge-decomposition, we have the following.

$$\begin{aligned}
\mathbb{E}\left[\mathcal{H}(\mathbf{x}^*) - \mathcal{H}(\widehat{\mathbf{x}^*})\right] &\leq \mathbb{E}\left[\sum_{(i,j)\in\mathcal{B}} (\psi_{ij}^U - \psi_{ij}^L)\right] \\
&= \sum_{(i,j)\in E} \Pr((i,j)\in\mathcal{B})(\psi_{ij}^U - \psi_{ij}^L) \\
&\leq \varepsilon\left[\sum_{(i,j)\in E} (\psi_{ij}^U - \psi_{ij}^L)\right] \\
&\leq \varepsilon(d^*+1)\mathcal{H}(\mathbf{x}^*). \qquad (18)
\end{aligned}$$

The running time bound can be obtained using arguments similar to those in Lemma 7. ■

C. Analysis of MODE*: Low doubling dimension G*

Here we interpret result obtained in Theorem 4 and Lemma 9, for G that has low doubling-dimension and uses decomposition scheme DB-DIM.

Theorem 5: Let MRF graph G of n nodes with doubling dimension ρ be given. Consider any $\varepsilon \in (0,1)$ such that $\rho(\rho + \log 1/\varepsilon) = o(\log\log n)$, define $\varphi = \varepsilon 2^{-\rho-3}$. Then MODE using DB-DIM$(\varphi, K(\varphi,\rho))$ produces bounds $\widehat{\mathbf{x}^*}$ such that

$$\mathbb{E}\left[\mathcal{H}(\mathbf{x}^*) - \mathcal{H}(\widehat{\mathbf{x}^*})\right] \leq \varepsilon\mathcal{H}(\mathbf{x}^*).$$

The algorithm takes $O(n2^\rho C_0(\varepsilon,\rho))$ time to obtain the estimate, where $C_0(\varepsilon,\rho) = |\Sigma|^{K(\varphi,\rho)^{2\rho}}$. Further, if $\rho(\rho + \log 1/\varepsilon) = o(\log\log n)$ then the algorithm takes $o(n^{1+\delta})$ amount of time for any $\delta > 0$.

Proof: Theorem 4, Lemma 9 and Lemma 2 imply that the output produced by MODE algorithm is such that

$$\begin{aligned}
\mathbb{E}\left[\mathcal{H}(\mathbf{x}^*) - \mathcal{H}(\widehat{\mathbf{x}^*})\right] &\leq \varepsilon 2^{-\rho-2}(d^*+1)\mathcal{H}(\mathbf{x}^*) \\
&\leq \varepsilon\mathcal{H}(\mathbf{x}^*), \qquad (19)
\end{aligned}$$

because $d^* + 1 \leq 2^{\rho+2}$ for a graph with doubling dimension ρ. The running time analysis of the algorithm follows exactly the same arguments as those in the proof of Theorem 2. ■

D. Analysis of MODE*: Minor-excluded G*

We apply Theorem 4 and Lemma 9 for minor-excluded graphs when the DECOMP procedure is the MINOR-E. We obtain the following precise result.

Theorem 6: Let MRF graph G of n nodes exclude $K_{r,r}$ as its minor. Let d^* be the maximum vertex degree in G. Given $\varepsilon > 0$, use MODE algorithm with MINOR-E(G, r, Λ) where $\Lambda = \lceil \frac{r(d^*+1)}{\varepsilon} \rceil$. Then,

$$\mathbb{E}\left[\mathcal{H}(\mathbf{x}^*) - \mathcal{H}(\widehat{\mathbf{x}^*})\right] \leq \varepsilon\mathcal{H}(\mathbf{x}^*).$$

Further, algorithm takes $(nC(d^*, |\Sigma|, \varepsilon))$, where $C(d^*, |\Sigma|, \varepsilon) = d^*|\Sigma|^{d^{*O(\Lambda)}}$. Therefore, if $\varepsilon^{-1}d^*\log d^* = o(\log\log n)$, then the algorithm takes $o(n^{1+\delta})$ steps for arbitrary $\delta > 0$.

Proof: From Lemma 5 about the MINOR-E algorithm, we have that with choice of $\Lambda = \lceil \frac{r(d^*+1)}{\varepsilon} \rceil$, the algorithm produces (ε, Δ) edge-decomposition where $\Delta = d^{*O(\Lambda)}$. Since its an (ε, Δ) edge-decomposition, from Lemma 9 it follows that

$$\mathbb{E}\left[\mathcal{H}(\mathbf{x}^*) - \mathcal{H}(\widehat{\mathbf{x}^*})\right] \leq \varepsilon\mathcal{H}(\mathbf{x}^*).$$

Now, by Lemma 9 the algorithm running time is $O(nd^*|\Sigma|^\Delta) + T_{\text{DECOMP}}$. As discussed earlier in Lemma 5, the algorithm MINOR-E takes $O(r|E|) = O(nrd^*)$ operations. That is, $T_{\text{DECOMP}} = O(nrd^*)$. Now, $\Delta = d^{*O(\Lambda)}$ and $\Lambda \leq r(d^*+1)/\varepsilon + 1$. Therefore, the first term of the computation time bound is bounded above by

$$O\left(nd^*|\Sigma|^{d^{*O(rd^*/\varepsilon)}}\right).$$

Now, we will establish that the above term is $O(n^2)$ under the hypothesis $\varepsilon^{-1}d^*\log d^* = o(\log\log n)$. The hypothesis implies that (since r a constant, not scaling with n):

$$\Lambda\log d^* = o(\log\log n).$$

That is, for any finite L (say, $L = 2$) we have that

$$\Delta = O(\log^{1/L} n).$$

This in turn implies that, for finite $|\Sigma|$ we have

$$|\Sigma|^\Delta = o(n^{\delta/2}),$$

for any $\delta > 0$. Since $d^* = o(\log\log n) = O(n^{\delta/2})$. Therefore, it follows that

$$O\left(nd^*|\Sigma|^\Delta\right) = o(n^{1+\delta}).$$

This completes the proof of Theorem 6. ■

VI. MESSAGE-PASSING IMPLEMENTATION THROUGH SELF-AVOIDING WALK

The approximate inference algorithms, LOG PARTITION and MODE presented above are *local* in the sense that in order to make computation, the centralization of the algorithm is limited only up to each connected component. This section provides a method for designing message-passing implementation for computing these estimates using the self-avoiding walk trees. This message passing algorithm is explained for MAP computation and is restricted to binary MRF. It is worth noting that any MAP estimation problem over discrete pair-wise exponential family can be converted into a binary pair-wise MRF with the help of addition nodes. This is explained in Appendix B. Thus, in principle, this message passing algorithm can work for any discrete valued Markov random field represented by a factor graph.

A. Equivalence: MRF and Self-Avoiding Walk Tree

The first result is about equivalence of max-marginal of a node, say v, in an MRF G and max-marginal of root of self-avoiding walk tree with respect to v. Dror Weitz [18] showed such equivalence in the context of marginal distributions of the nodes. We establish the result for max-marginal. However, the proof is a direct adaption of the proof of result by Weitz [18].

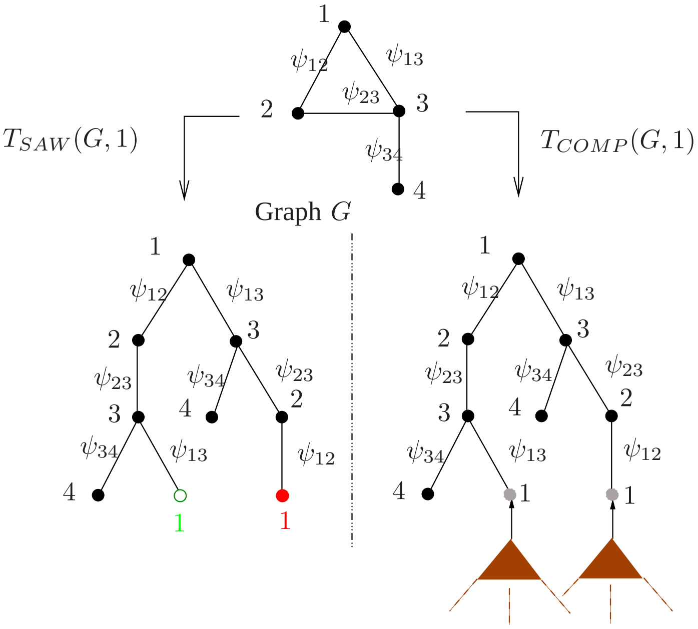

Fig. 3. A graph G of 4 nodes with one loop is given. On left, we have the self-avoiding walk tree of G for node 1, i.e. $T_{SAW}(G,1)$ with green and red being special nodes. On right, we have computation tree $T_{COMP}(G,1)$ for node 1's computation under Belief Propagation (or Max-Product) algorithm. The grey nodes of $T_{COMP}(G,1)$ correspond to green and red node of $T_{SAW}(G,1)$ on the left.

Given binary pair-wise MRF G of n nodes, our interest is in finding

$$p_v^*(\gamma) = \max_{\sigma\in\{0,1\}^n:\sigma_v=\gamma} \Pr(\sigma), \text{ for } \gamma \in \{0,1\} \text{ for all } v.$$

Definition 1 (Self-Avoiding Walk Tree): Consider graph $G = (V,E)$ of pair-wise binary MRF. For $v \in V$, we define the self avoiding walk tree $T_{SAW}(G,v)$ as follows. First, for each $u \in V$, give an ordering of its neighbors $N(u)$. This ordering can be arbitrary but remains fixed forever. Given this, $T_{SAW}(G,v)$ is constructed by the breadth first search of nodes of G starting from v without backtracking. Then stop the bread-first search along a direction when an already visited vertex is encountered (but include it in $T_{SAW}(G,v)$ as a leaf). Say one such leaf be $\hat{w}$ of $T_{SAW}(G,v)$ and let it be a copy of a node w in G. We call such a leaf node of $T_{SAW}(G,v)$ as *Marked*. A marked leaf node is assigned color *Red* or *Green* according to the following condition: The leaf $\hat{w}$ is marked since we encountered node w of G twice along our bread-first search excursion. Let the (directed) path between these two encounters of w in G be given by $(w, v_1, \dots, v_k, w)$. Naturally, $v_1, v_k \in N(w)$ in G. We mark the leaf node $\hat{w}$ as *Green* if according to the ordering done by node w in G of its neighbors, if v_k is given smaller number than that of v_1. Else, we mark it as *Red*. Let $\mathbf{V}_v$ and $\mathbf{E}_v$ denote the set of nodes and vertices of tree $T_{SAW}(G,v)$. With little abuse of notation, we will call root of $T_{SAW}(G,v)$ as v.

Given a $T_{SAW}(G,v)$ for a node $v \in V$ in G, an MRF is naturally induced on it as follows: all edges inherit the pair-wise compatibility function (i.e. $\psi_{\cdot\cdot}(\cdot,\cdot)$) and all nodes inherit node-potentials (i.e. $\phi_\cdot(\cdot)$) from those of MRF G in a natural manner. The only distinction is the modification of the node-potential of *marked* leaf nodes of $T_{SAW}(G,v)$ as follows. A marked leaf node, say $\hat{w}$ of $T_{SAW}(G,v)$ modifies its potentials as follows: if it is *Green* than it sets $\phi_{\hat{w}}(1) = \phi_w(1), \phi_{\hat{w}}(0) = 0$ but if it is *Red* leaf node then it sets $\phi_{\hat{w}}(0) = \phi_w(0), \phi_{\hat{w}}(1) = 0$.

Example 1 (Self-avoiding walk tree): Consider 4 node binary pair-wise MRF G in Figure 3. Let node 1 gives number a to node 2, number b to node 3 so that $a > b$. Given this numbering, the bottom left of Figure 3 represents $T_{SAW}(G,1)$. The Green leaf node essentially means that we set its value permanently to 1.

With above description, $T_{SAW}(G,v)$ gives rise to a pair-wise binary MRF. Let $\mathbb{Q}_{G,v}$ denote the probability distribution induced by this MRF on boolean cube $\{0,1\}^{|\mathbf{V}_v|}$. Our interest will be in the max-marginal for root v or equivalently

$$q_v^*(\gamma) = \max_{\sigma\in\{0,1\}^{|\mathbf{V}_v|}:\sigma_v=\gamma} \mathbb{Q}_{G,v}(\sigma), \text{ where } \gamma \in \{0,1\}.$$

Here we present an equivalence between $p_v^*(\cdot)$ and $q_v^*(\cdot)$. This is a direct adaptation of result by Weitz [18].

Theorem 7: Consider any binary pair-wise MRF $G = (V,E)$. For any $v \in V$, let $p_v^*(\cdot)$ be as defined above with respect to $\Pr_G$. Let $T_{SAW}(G,v)$ be the self-avoiding walk tree MRF and let $q_v^*(\cdot)$ be as defined above for root node of $T_{SAW}(G,v)$ with respect to $\mathbb{Q}_{G,v}$. Then,

$$\frac{p_v^*(1)}{p_v^*(0)} = \frac{q_v^*(1)}{q_v^*(0)}. \tag{20}$$

Here we allow ratio to be $0, \infty$.

Proof: The proof follows by induction. As a part of the proof, we will come across graphs with some *fixed* vertices, where a vertex u is said to be fixed to 0 (resp. 1) if $\phi_u(0) > 0$, $\phi_u(1) = 0$ (resp. $\phi_u(1) > 0$, $\phi_u(0) = 0$). The induction is on the number of *unfixed* vertices of G. We essentially prove the following, which implies the statement of Lemma: given any pair-wise MRF on a graph G (with possibly some *fixed* vertices), construct corresponding $T_{SAW}(G,v)$ MRF for some node v. If the number of *unfixed* vertex of G is at most m, then the (20) holds. Next, inductive proof.

Initial condition. Trivially the desired statement holds for any graph with exactly one *unfixed* vertex, by definition of MRF, i.e. (1). The reason is that for such a graph, due to all but one node being fixed, the max-marginal of each node is purely determined by its immediate neighbors due to Markovian nature of MRF. The immediate neighborhood of v in $T_{SAW}(G,v)$ and G is the same.

Hypothesis. Assume that the statement is true for any graph with less than or equal to $m \in \mathbb{N}$ *unfixed* nodes.

Induction step. Without loss of generality, suppose that our graph of interest, G, has $m+1$ *unfixed* vertices. If v is a *fixed* vertex, then (20) holds trivially. Let $v \in V$ be an unfixed vertex of G. Then we will show via inductive hypothesis that

$$\frac{q_v^*(1)}{q_v^*(0)} = \frac{p_v^*(1)}{p_v^*(0)}.$$

Let d be the degree of v; $v_1, v_2, \ldots, v_d$ be the neighbors of v where the order of neighbors is the same as that used in definition of $T_{SAW}(G,v)$. Let T_ℓ be the ℓth subtree of $T_{SAW}(G,i)$ having v_ℓ as its root and $Y(\ell)$ be the binary pair-wise MRF induced on T_ℓ by restriction of $T_{SAW}(G,v)$. Let $q_\ell^*(\sigma)$ be the max-marginal of vertex v_ℓ taking value $\sigma \in \Sigma = \{0,1\}$ with respect to $Y(\ell)$. Note that when T_ℓ consists of a single vertex, then $q_\ell^*(\sigma) \propto \phi_{v_\ell}(\sigma)$. Let $\lambda_v = \frac{\phi_v(1)}{\phi_v(0)}$. Then from definition of pair-wise MRF and tree-structure,

$$\frac{q_v^*(1)}{q_v^*(0)} = \lambda_v \prod_{\ell=1}^{d} \frac{\max_{\sigma\in\Sigma} \psi_{v_\ell,v}(\sigma,1) q_\ell^*(\sigma)}{\max_{\sigma\in\Sigma} \psi_{v_\ell,v}(\sigma,0) q_\ell^*(\sigma)}. \qquad (21)$$

Now to calculate $\frac{p_v^*(1)}{p_v^*(0)}$, we define a new graph G' and the corresponding pair-wise MRF X' as follows. Let G' be the same as G except that v is replaced by d vertices $v'_1, v'_2, \ldots, v'_d$; each v'_ℓ is connected only to v_ℓ, $1 \le \ell \le d$. The X' is defined same as X except that $\phi_{v'_\ell}(1) = \lambda_v^{1/d}\phi_v(1)$, $\phi_{v'_\ell}(0) = \phi_v(0)$ and $\psi_{v_\ell v'_\ell} = \psi_{v_\ell v}$. Then,

$$\frac{p_v^*(1)}{p_v^*(0)} = \frac{\max_{\left\{X' : X'_{v'_1}=1, X'_{v'_2}=1, \ldots, X'_{v'_d}=1\right\}} \Pr_{G'}(X')}{\max_{\left\{X' : X'_{v'_1}=0, X'_{v'_2}=0, \ldots, X'_{v'_d}=0\right\}} \Pr_{G'}(X')} = \prod_{\ell=1}^{d} \frac{\mu_\ell(1)}{\mu_\ell(0)}, \qquad (22)$$

where define $\mu_\ell(\sigma) = \max_{\{X' : X'_{v'_\ell}=\sigma\}} \Pr[X' | \; X'_{v'_1} = 0, \ldots, X'_{v'_{(\ell-1)}} = 0, X'_{v'_{(\ell+1)}} = 1, \ldots, X'_{v'_d} = 1]$. The second equality in (22) follows by standard trick of Telescoping multiplication and Lemma 10.

Now for $1 \le \ell \le d$, consider MRF $X'(\ell)$ induced on $G'(\ell) = G' - \{v'_\ell\}$ by fixing $\{v'_1, \ldots v'_d\} - \{v'_\ell\}$ as follows: let $(\phi_{v'_1}(0) = 1, \phi_{v'_1}(1) = 0); \ldots; (\phi_{v'_{\ell-1}}(0) = 1, \phi_{v'_{\ell-1}}(1) = 0); (\phi_{v'_{\ell+1}}(0) = 0, \phi_{v'_{\ell+1}}(1) = 1); \ldots; (\phi_{v'_d}(0) = 0, \phi_{v'_d}(1) = 1)$. Then let $\nu_\ell(\sigma), \sigma \in \Sigma$ denote the max-marginal of v_ℓ for taking value σ with respect to $X'(\ell)$. Given this, by definition of MRF X' as well $X'(\ell)$ and noting that v'_ℓ is a leaf (only connected to v_ℓ) with respect to graph G', we have

$$\frac{\mu_\ell(1)}{\mu_\ell(0)} = \lambda_v^{1/d} \frac{\max_{\sigma\in\Sigma} \psi_{v_\ell,v'_\ell}(\sigma,1)\nu_\ell(\sigma)}{\max_{\sigma\in\Sigma} \psi_{v_\ell,v'_\ell}(\sigma,0)\nu_\ell(\sigma)}. \qquad (23)$$

From (21), (22) and (23) it is sufficient to show that

$$\frac{\nu_\ell(1)}{\nu_\ell(0)} = \frac{q_\ell^*(1)}{q_\ell^*(0)}, \quad 1 \le \ell \le d. \qquad (24)$$

Now, note that T_ℓ is the same as $T_{SAW}(G(\ell))$ with respect to $X'(\ell)$. Because for each $\ell = 1, \ldots d$, $G'(\ell)$ has one less *unfixed* node than G, the desired result (24) follows by induction hypothesis. ■

Lemma 10: Consider a distribution on $X = (X_1, \ldots, X_n)$ where X_i are binary variables. Let $p_s = \Pr[X = s], s \in \Sigma^n$. Let $p_{s|a_2,\ldots,a_d} = \Pr[X = s | X_2 = a_2, \ldots, X_d = a_d]$ for any $d \ge 1$. Let $S(a_1, \ldots, a_d) = \{s = (s_1, \ldots, s_n) \in \Sigma^n : s_1 = a_1, \ldots, s_d = a_d\}$. Then,

$$\frac{\max_{s\in S(a_1,a_2\ldots,a_d)} p_s}{\max_{s\in S(\hat{a}_1,a_2,\ldots,a_d)} p_s} = \frac{\max_{s\in S(a_1,a_2\ldots,a_d)} p_{s|a_2,\ldots,a_d}}{\max_{s\in S(\hat{a}_1,a_2,\ldots,a_d)} p_{s|a_2,\ldots,a_d}}.$$

Proof: Let $q = \Pr(X_2 = a_2, \ldots, X_d = a_d)$. Then, by definition of conditional probability for $s \in S(a_1, a_2, \ldots, a_d) \cup S(\hat{a}_1, a_2, \ldots, a_d)$, $p_s = p_{s|a_2,\ldots,a_d} q$. From this, Lemma follows immediately. ■

B. Size of Self-avoiding walk tree

We present a novel characterization of the size of the self avoiding walk tree in terms of number of edges in it (which is equal to number of nodes minus 1). This characterization is necessary to obtain bound on the running time of the self-avoiding walk tree. This combinatorial result should be of interest in its own right.

Lemma 11: Consider a connected graph $G = (V, E)$ with $|V| = n$ nodes and $|E| = n - 1 + k$ edges, $k \ge 0$. Then for any $v \in V$, $|T_{SAW}(G,v)| \le (n + k - 1)2^{k+1}$. Further, there exists a graph with $n - 1 + k$ edges with $k < n/2$ so that for any node $v \in V$, $|T_{SAW}(G,v)| \ge n2^{k-2}$.

Proof: The proof is divided into two parts. We first provide the proof of lower bound. Consider a line graph of n nodes (with $n-1$ edges). Now add $k < n/2$ edges as follows. Add an edge between 1 and n. Remaining $k - 1$ edges are added between node pairs: $(2,4), (4,6), \ldots, (2(k-2), 2(k-1)), (2(k-1), 2k)$. Consider any node, say v. It is easy to see that there are at least 2^{k-2} different ways in which one can start walking on the graph from node v towards node 1, cross from 1 to n via edge $(1, n)$ and then come back to node v. Each of these different loops, starting from v and ending at v creates 2 distinct paths in the self-avoiding walk tree of length at least$\frac{n}{2}$. Thus, the size of self-avoiding walk tree of each node is at least $n2^{k-2}$ for each node. This completes the proof of lower bound.

Now, we prove the upper bound of $n2^{k+1}$ on the size of self-avoiding walk tree for each node $v \in V$. Given that G is connected, we can divide the edge set $E = E_T \cup E_k$ where $E_k = \{e_1, \ldots, e_k\}$ and $T = (V, E_T)$ forms a spanning tree of G. Let $\mathcal{S}$ be the set of all subsets of $E_k = \{e_1, \ldots, e_k\}$ (there are 2^k of them including empty set). Now fix a vertex $v \in V$ and we will concentrate on $T_{SAW}(G,v)$. Consider any $u \in V$ (can be v) and $S \in \mathcal{S}$. Next, we wish to count number of paths in $T_{SAW}(G,v)$ that end at (a copy of) u (however, u need not be a leaf), contain all edges in S but none from $E_k \backslash S$. We claim the following.

Claim. There can be at most one path of $T_{SAW}(G,v)$ from v to (a copy of) u and containing all edges from S but none from $E_k \backslash S$.

Proof: To prove the above claim, suppose it is not true. Then there are at least two distinct paths from v to u that contain all edges in S (but none from $E_k \backslash S$). Consider the symmetric difference of these two paths (in terms of edges). This symmetric difference must be a non-empty subset of E_T and also contain a loop (as the two paths have same starting and ending point). But this is not possible as $T = (V, E_T)$

is a tree and it does not contain a loop. This contradicts our assumption and proves the claim. ■

Given the above claim, for any node u, clearly the number of distinct paths from node v to (a copy of) u in $T_{SAW}(G, v)$ are at most 2^k. Now each edge has two end points. For each appearance of an edge of G in $T_{SAW}(G, v)$, a distinct path from v to one of its end point must appear in $T_{SAW}(G, v)$. From above claim, this can happen at most $2 \times 2^k = 2^{k+1}$. There are $n + k - 1$ edges of G in total. Thus, net number of edges that can appear in $T_{SAW}(G, v)$ is at most $(n + k - 1)2^{k+1}$; thus completing the proof of Lemma 11. ■

C. Algorithm: At a higher level

Now, we describe algorithm to compute MAP approximately. The algorithm is the same as MODE, however computation restricted to each component is done through self-avoiding walk. Specifically, the algorithm does the following: given G, decompose it into (small) components $S_1, \dots, S_K$ by removing (few) edges $\mathcal{B} \subset E$, where $\mathcal{B}$ is obtained using DECOMP; (as before, for minor-excluded graph use MINOR-E and DB-DIM for graphs with low doubling dimension). Then, compute an approximate MAP assignment by computing exact MAP restricted to the components. This exact computation for each component is performed through a message passing mechanism using the equivalence stated in Theorem 7: essentially, growing self-avoiding walk tree is just sending messages along a breadth-first search tree; computation over a self-avoiding walk tree is essentially standard max-product (message passing) algorithm. The precise schedule for message-passing is described in the next sub-section. Here, we describe algorithm for any graph G at a higher-level.

```
MODE($G$)
(1) Use DECOMP($G$) to obtain $\mathcal{B} \subset E$ such that
    (a) $G' = (V, E \backslash \mathcal{B})$ is made of connected components $S_1, \dots, S_K$.
(2) For each connected component $S_j, 1 \leq j \leq K$, do the following:
    (a) Compute exact MAP $\mathbf{x}^{*,j}$ for component $S_j$, where $\mathbf{x}^{*,j} = (x_i^{*,j})_{i \in S_j}$.
    (b) Computation of $\mathbf{x}_i^{*,j}$ is performed by growing self-avoiding walk tree at node $i$ restricted to induced graph by nodes of $S_j$ using a message passing mechanism; then computing max-marginal on self-avoiding walk tree using message passing mechanism (i.e. standard max-product algorithm on self-avoiding walk tree).
(3) Produce output $\widehat{\mathbf{x}^*}$, which is obtained by assigning values to nodes using $\mathbf{x}^{*,j}, 1 \leq j \leq K$. This is clearly local operation.
```

D. Algorithm: Message-passing schedule

The following is a pseudo-code of a distributed message passing algorithm MSG-PASS-MODE which computes $\mathbf{x}^{*,j}$ for each component S_j. The MSG-PASS-MODE finds exact MAP, by Theorem 7. This section is of interest primarily for the reason that it provides the detailed distributed message-passing implementation for computing MAP. A reader, not interested in such detailed implementation, may skip this section.

To describe the pseudo-code, we need some notation. Each node $v \in V$, let $N(v)$ denote the set of all its neighbors, i.e. $N(v) = \{u \in V : (u, v) \in E\}$. Node v assigns an arbitrary fixed order to all nodes in $N(v)$. For example, if v has neighbors u, w and z then it can number u as the first neighbor, w as second neighbor and z as third neighbor. The ordering chosen by each node is independent of choices of all other nodes. The algorithm operates in two phases. In the first phase, algorithm explores local topology for each node via sending "path sequences". By "path sequence" we mean a finite sequence of vertices $(v_1, v_2, \dots, v_k)$, where $(v_\ell, v_{\ell+1}) \in E$ for $1 \leq \ell \leq k - 1$. In the second phase, algorithm uses the path sequences to recursively calculate "computation sequence" which in turn leads to calculation of $q_v^*(\cdot)$ at nodes. A "computation sequence" is of the form $(v_1, v_2, \dots, v_k, m_{v_k}(0), m_{v_k}(1))$, where $m_{v_k}(\cdot)$ are certain real-numbers (which have interpretation of message). As we shall see, the structure of recursive calculation to obtain "computation sequence" is the same as that of max-product algorithm. Thus, there is very strong connection between MP and MSG-PASS-MODE. For ease of exposition, the algorithm is described to compute the ratio $q_v^*(1)/q_v^*(0)$ for all $v \in V$.

```
MSG-PASS-MODE(G)
(0) Initially, each vertex $v$ sends a path sequence $(v)$ to each of its neighbors.
(1) When node $u$ receives a path sequence $(v_1, v_2, \dots, v_k)$ from its neighbor $v$, (note that, by construction given later, $v_k = v$) it does the following:
    ○ If $u$ is a leaf (i.e. $u$ is connected only to $v$), $u$ sends back a computation sequence $(v_1, v_2, \dots, v_k, u, m_u(0), m_u(1))$ to $v$, where
          $m_u(\sigma) \propto \max_{\sigma_u \in \Sigma} \psi_{u,v}(\sigma_u, \sigma)\phi_u(\sigma_u)$
          $\sum_{\sigma_u \in \Sigma} m_u(\sigma_u) = 1.$        (25)
    ○ If $u$ is not a leaf, check whether $u$ appears among $v_\ell, 1 \leq \ell \leq k$:
        ∗ If NO, $u$ sends a path sequence $(v_1, \dots, v_k, u)$ to each of $u$'s neighbors but $v$.
        ∗ If YES, then let $v_\ell = u, 1 \leq \ell < k$.
          – If, with respect to the ordering given by node $u$ to its neighbors, the rank (order) of node $v_{\ell+1}$ is larger then $v$, then $u$ sends back (to $v$) a computation sequence $(v_1, v_2, \dots, v_k, u, m_u(0), m_u(1))$, where $m_u(1) = 1$ and $m_u(0) = 0$.
          – Otherwise (i.e. the rank of node $v_{\ell+1}$ is smaller than $v$), $u$ sends back (to $v$) computation sequence $(v_1, v_2, \dots, v_k, u, m_u(0), m_u(1))$, where $m_u(0) = 1$ and $m_u(1) = 0$.
(2) Once a node $u$ receives a computation sequence $(v_1, \dots, v_k, m_{v_k}(0), m_{v_k}(1))$ from its neighbor $v$, (note
```

that, by construction $v_k = v$ and $v_{k-1} = u$). Store this computation sequence in u's memory and do the following:

- ○ If $k > 2$, check whether u has stored computation sequences of the form $(v_1, \dots, v_{k-1}, w, m_w(0), m_w(1))$ for all $w \in N(u) - \{v_{k-2}\}$. If so, u sends a computation sequence $(v_1, \dots, v_{k-1}(= u), m_u(0), m_u(1))$ to v_{k-2} where

$$m_u(\sigma) \propto \left[\max_{\sigma_u \in \Sigma} \psi_{u,v_{k-2}}(\sigma_u, \sigma)\phi_u(\sigma_u) \times \prod_{w \in N(u)-\{v_{k-2}\}} m_w(\sigma_u)\right],$$

$$\sum_{\sigma_u \in \Sigma} m_u(\sigma_u) = 1.$$

  Delete computation sequences $(v_1, \dots, v_{k-1}, w, m_w(0), m_w(1))$ for all $w \in N(j) - \{i_{k-2}\}$ from u's memory.
- ○ If $k = 2$, then check whether for all $w \in N(j)$, u has stored computation sequences $(v_1, w, m_w(0), m_w(1))$. If so, compute the (estimate of) max-belief of u as

$$q_u^*(\sigma) \propto \phi_u(\sigma) \prod_{w \in N(u)} m_w(\sigma), \quad \text{and} \quad \sum_{\sigma \in \Sigma} q_u^*(\sigma) = 1.$$

(3) When all nodes have computed their max-beliefs, declare $q_v^*(1)/q_v^*(0)$ as an estimate of $p_v^*(1)/p_v^*(0)$ $\forall\, v \in V$.

VII. Experiments

Our algorithm provides provably good approximation for any MRF that has low doubling dimension or that excluded minor. The planar graph is a special case of such graphs. The popular model of grid graph, which is both planar and has low doubling dimension, will be used in the experimental section. We will, however, use the decomposition algorithm MINOR-E for obtaining our results. Now we present detailed setup and experimental results.

A. Setup 1

Consider[2] binary (i.e. $\Sigma = \{0, 1\}$) MRF on an $n \times n$ lattice $G = (V, E)$:

$$\Pr(\mathbf{x}) \propto \exp\left(\sum_{i \in V} \theta_i x_i + \sum_{(i,j) \in E} \theta_{ij} x_i x_j\right), \quad \text{for } \mathbf{x} \in \{0, 1\}^{n^2}.$$

Figure 4 shows a lattice or grid graph with $n = 4$ (on the left side). There are two scenarios for choosing parameters (with notation $\mathcal{U}[a, b]$ being uniform distribution over interval $[a, b]$):

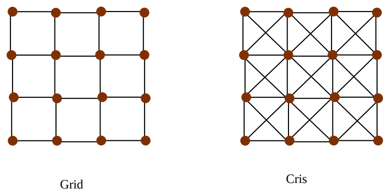

Fig. 4. Example of grid graph (left) and cris-cross graph (right) with $n = 4$.

(1) *Varying interaction.* θ_i is chosen independently from distribution $\mathcal{U}[-0.05, 0.05]$ and θ_{ij} chosen independent from $\mathcal{U}[-\alpha, \alpha]$ with $\alpha \in \{0.2, 0.4, \dots, 2\}$.

(2) *Varying field.* θ_{ij} is chosen independently from distribution $\mathcal{U}[-0.5, 0.5]$ and θ_i chosen independently from $\mathcal{U}[-\alpha, \alpha]$ with $\alpha \in \{0.2, 0.4, \dots, 2\}$.

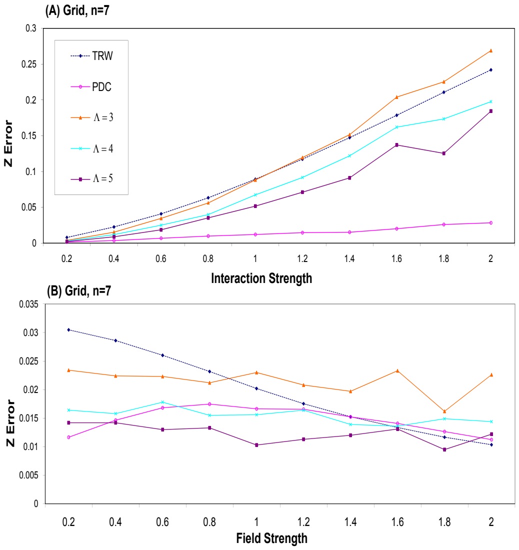

Fig. 5. Comparison of TRW, PDC and our algorithm for grid graph with $n = 7$ with respect to error in $\log Z$. Our algorithm outperforms TRW and is competitive with respect to PDC.

The grid graph is planar. Hence, we run our algorithms LOG PARTITION and MODE, with decomposition scheme MINOR-E$(G, 3, \Lambda)$, $\Lambda \in \{3, 4, 5\}$. We consider two measures to evaluate performance: error in $\log Z$, defined as $\frac{1}{n^2}|\log Z^{\text{alg}} - \log Z|$; and error in $\mathbb{E}(\mathbf{x}^*)$, defined as $\frac{1}{n^2}|\mathbb{E}(\mathbf{x}^{\text{alg}} - \mathbb{E}(\mathbf{x}^*)|$.

We compare our algorithm for error in $\log Z$ with the two recently very successful algorithms – Tree re-weighted algorithm (TRW) and planar decomposition algorithm (PDC). The comparison is plotted in Figure 5 where $n = 7$ and results are averages over 40 trials. The Figure (A) plots error with respect to varying interaction while Figure (B) plots error with respect to varying field strength. Our algorithm, essentially outperforms TRW for these values of Λ and perform very competitively with respect to PDC.

The key feature of our algorithm is scalability. Specifically, running time of our algorithm with a given parameter value Λ scales linearly in n, while keeping the relative error bound exactly the same. To explain this important feature, we plot the theoretically evaluated bound on error in $\log Z$ in Figure 6 with

[2] Though this setup has ϕ_i, ψ_{ij} taking negative values, they are equivalent to the setup considered in the paper as the function values are lower bounded and hence *affine* shift will make them non-negative without changing the distribution.

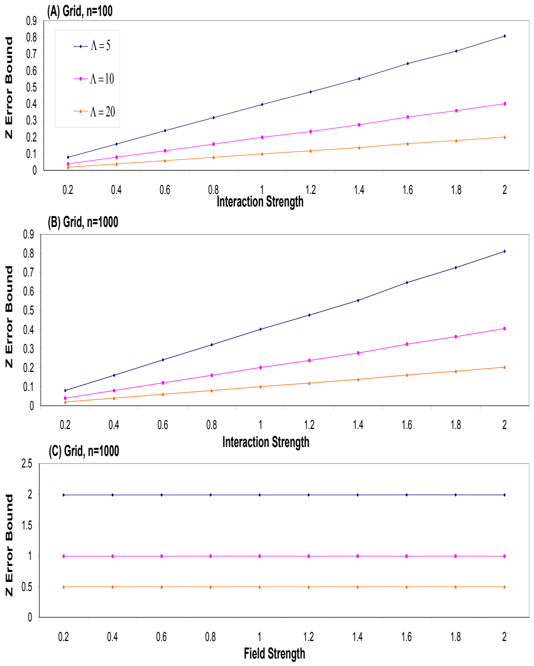

Fig. 6. The theoretically computable error bounds for $\log Z$ under our algorithm for grid with $n = 100$ and $n = 1000$ under varying interaction and varying field model. This clearly shows scalability of our algorithm.

tags (A), (B) and (C). Note that error bound plot is the same for $n = 100$ (A) and $n = 1000$ (B). Clearly, actual error is likely to be smaller than these theoretically plotted bounds. We note that these bounds only depend on the interaction strengths and *not* on the values of fields strengths (C).

Results similar to of LOG PARTITION are expected from MODE. We plot the theoretically evaluated bounds on the error in MAP in Figure 7 with tags (A), (B) and (C). Again, the bound on MAP relative error for given Λ parameter remains the same for all values of n as shown in (A) for $n = 100$ and (B) for $n = 1000$. There is no change in error bound with respect to the field strength (C).

B. Setup 2

Everything is exactly the same as the above setup with the only difference that grid graph is replaced by *cris-cross* graph which is obtained by adding extra four neighboring edges per node (exception of boundary nodes). Figure 4 shows cris-cross graph with $n = 4$ (on the right side). We again run the same algorithm as above setup on this graph. For cris-cross graph, which is graph with low-doubling dimension, we obtained its graph decomposition from the decomposition of its grid subgraph. Therefore , the running time of our algorithm remains the same (in order) as that of grid graph and error bound will become only 3 times weaker than that for the grid graph. We compute these theoretical error bounds for $\log Z$ and MAP which is plotted in Figure 8 and 9. These figures are similar to the Figures 6 and 7 for grid graph.

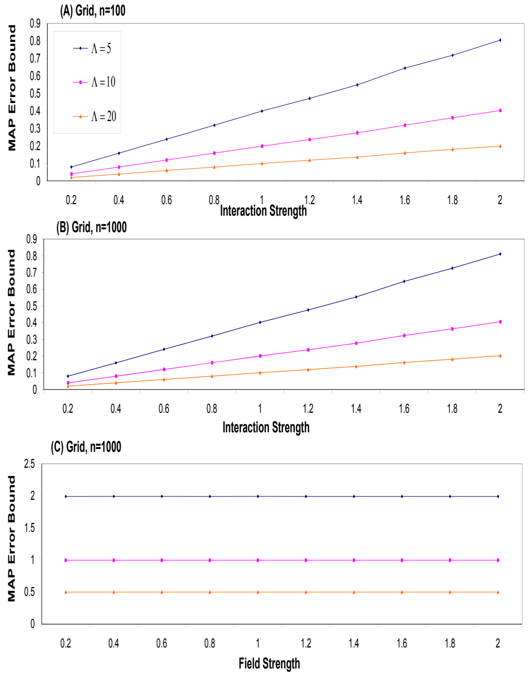

Fig. 7. The theoretically computable error bounds for MAP under our algorithm for grid with $n = 100$ and $n = 1000$ under varying interaction and varying field model.

VIII. UNEXPECTED IMPLICATION: EXISTENCE OF LIMIT

This section describes an important and somewhat unexpected implication of our results, specifically Lemmas 7 and 9. In the context of *regular* MRF, such as an MRF on $\mathbb{Z}_n^d$ (of n^d nodes) with same node and edge potential functions for all nodes and edges, we will show that (non-trivial) limit $\frac{1}{n^d} \log Z$ exists as $n \to \infty$. It is worth noting that showing existence of such limits is not straightforward in general and hence our method should be of interest as such an analytic tool. We believe that the result stated below is well-known; however its proof method is likely to allow for establishing such existence for a more general class of problems. As an example, the theorem will hold even when node and edge potentials are not the same but are chosen from a class of such potential as per some distribution in an i.i.d. fashion. Now, we state the result.

Theorem 8: Consider a regular MRF of n^d nodes on d-dimensional grid $\mathbb{Z}_d^n = (V_n, E_n)$: let $\psi_{ij} \equiv \psi, \phi_i \equiv \phi$ for all $i \in V_n, (i,j) \in E_n$ with $\psi : \Sigma^2 \to \mathbb{R}_+, \phi : \Sigma \to \mathbb{R}_+$. Let Z_n be partition function of this MRF. Then, the following limit exists:

$$\lim_{n \to \infty} \frac{1}{n^d} \log Z_n = A(d, \phi, \psi) \in (0, \infty).$$

A. Proof of Theorem 8

The proof of Theorem 8 is stated for $d = 2$ and $\Sigma = \{0, 1\}$ case. Proof for $d \geq 3$ and Σ with $|\Sigma| \geq 2$ can be proved using exactly the same argument. The proof will use the following Lemmas.

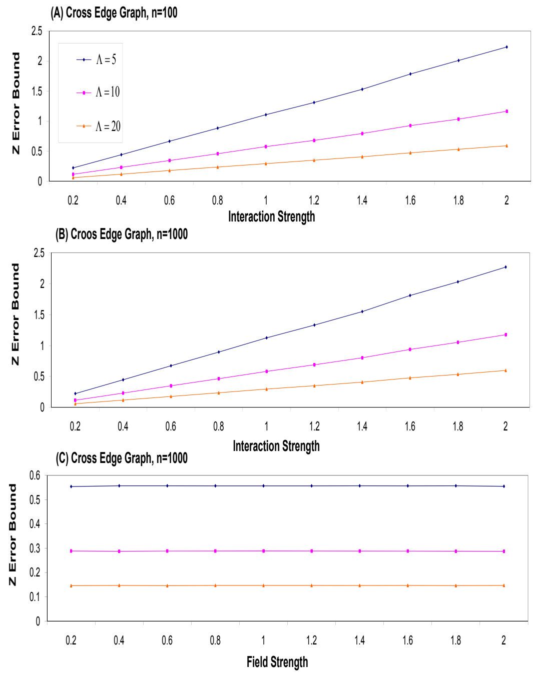

Fig. 8. The theoretically computable error bounds for $\log Z$ under our algorithm for cris-cross with $n = 100$ and $n = 1000$ under varying interaction and varying field model. This clearly shows scalability of our algorithm and robustness to graph structure.

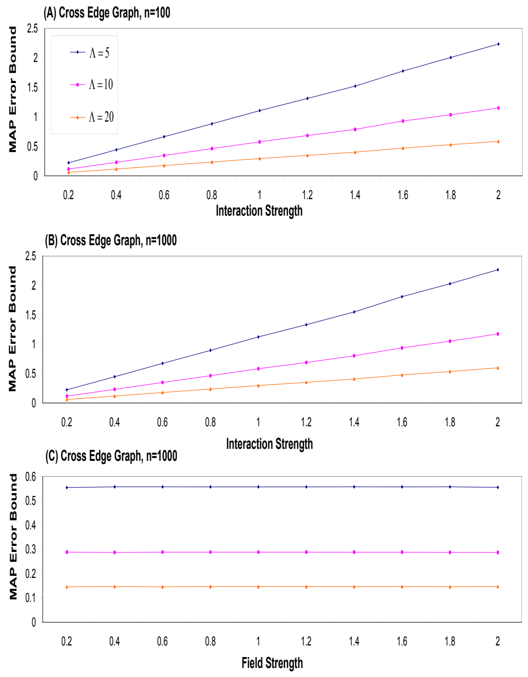

Fig. 9. The theoretically computable error bounds for MAP under our algorithm for cris-cross with $n = 100$ and $n = 1000$ under varying interaction and varying field model.

Lemma 12: Let $d = 2$ and $\phi^* = \max_{\sigma \in \{0,1\}} \phi(\sigma)$, $\psi^* = \max_{(\sigma,\sigma') \in \{0,1\}^2} \psi(\sigma, \sigma')$. Then,

$$n^2 \le \log Z_n \le \alpha n^2,$$

where $\alpha = \log 2 + \log \phi^* + 4 \log \psi^*$.

Lemma 13: Define $a_n = \frac{1}{n^2} \log Z_n$. Now, given $k > 0$, there exists $n(k)$ large enough such that for any $m, n \ge n(k)$,

$$|a_m - a_n| = O\left(\frac{1}{k}\right) + O\left(\frac{k}{\min\{m, n\}}\right).$$

Proof: (Theorem 8) We state proof of Theorem 8, before proving the above stated Lemmas. First note that, by Lemma 12, the elements of sequence $a_n = n^{-2} \log Z_n$ take value in $[1, \alpha]$. Now, suppose the claim of theorem is false. That is, sequence a_n does not converge as $n \to \infty$. That is, there exists $\delta > 0$ such for any choice of n_0, there are $m \ge n \ge n_0$ such that

$$|a_m - a_n| \ge \delta.$$

By Lemma 13, we can select k large enough and later $n_0 \ge n(k)$ large enough such that for any $m, n \ge n_0$,

$$|a_m - a_n| < \delta.$$

But this is a contradiction to our assumption that a_n does not converge to a limit. That is, we have established that a_n converges to a non-trivial limit in $[1, \alpha]$ as desired. This completes the proof of Theorem 8. ■

B. Proofs of Lemmas

Proof: (Lemma 12) Consider the following.

$$\begin{aligned} 2^{n^2} &= \sum_{\mathbf{x} \in \{0,1\}^{n^2}} \prod_{i \in V_n} 1 \prod_{(i,j) \in E_n} 1 \\ &\overset{(a)}{\le} \sum_{\mathbf{x} \in \{0,1\}^{n^2}} \prod_{i \in V_n} \exp(\phi(x_i)) \prod_{(i,j) \in E_n} \exp(\psi(x_i, x_j)) \\ &= Z_n \\ &\overset{(b)}{\le} \sum_{\mathbf{x} \in \{0,1\}^{n^2}} \prod_{i \in V_n} \exp(\phi^*) \prod_{(i,j) \in E_n} \exp(\psi^*). \end{aligned} \tag{26}$$

Here, (a) follows from the fact that ψ, ϕ are non-negative valued functions and (b) follows from definitions of ϕ^*, ψ^*. Now, taking logarithm on both sides implies the Lemma 12. ■

Proof: (Lemma 13) Given $k > 0$, consider n large enough (will be decided later). Consider $\mathbb{Z}_n^2 = (V_n, E_n)$ and let it be laid out on $X - Y$ plane so that its node in V_n occupy the integral locations : (i, j), $0 \le i \le n - 1, 0 \le j \le n - 1$. Now, we describe a scheme to obtain a $(O(1/k), O(k^2))$ edge-decomposition of $\mathbb{Z}_n^2$. For this, choose $\ell_1, \ell_2 \in \{0, \ldots, k - 1\}$ independently and uniformly at random. Select edges to form $\mathcal{B}$ to obtain edge-decomposition as follows: select vertical edges with bottom vertex having Y coordinate $\ell_2 + jk, j \ge 0$, and select horizontal edges with left vertex having X coordinate $\ell_1 + jk, j \geq 0$. That is,

$$\begin{aligned}\mathcal{B} &= \{(u,v) \in E_n : u = (i,j), v = (i+1,j), i \text{ mod } k = \ell_1\} \\ &\cup \{(u,v) \in E_n : u = (i,j), v = (i,j+1), j \text{ mod } k = \ell_2\}.\end{aligned}$$

It is easy to check that this is $(O(1/k), O(k^2))$ edge-decomposition due to uniform selection of ℓ_1, ℓ_2 from $\{0, \ldots, k-1\}$. Therefore, by Lemma 7, we can obtain estimates that are $(1 \pm O(1/k)) \log Z_n$ using our algorithm.

Let $m = \lceil n/k \rceil$. Under the decomposition $\mathcal{B}$ as described above, there are at least $(m-1)^2$ connected components that are MRF on $\mathbb{Z}_k$. Also, all the connected components can be covered by at most $(m+1)^2$ identical MRFs on $\mathbb{Z}_k$. Using arguments similar to those employed in calculations of Theorem 1 (using non-negativity of ϕ, ψ), it can be shown that the estimate produced by our algorithm is lower bounded as

$$\begin{aligned}(1 - O(1/k))(m-1)^2 \log Z_k &= n^2 \frac{\log Z_k}{k^2} \times \\ &\quad (1 - O(1/k) - O(k/n)),\end{aligned}$$

and is upper bounded as

$$\begin{aligned}(1 + O(1/k))(m+1)^2 \log Z_k &= n^2 \frac{\log Z_k}{k^2} \times \\ &\quad (1 + O(k/n) + O(1/k)).\end{aligned}$$

Therefore, from above discussion we obtain that

$$\frac{1}{n^2} \log Z_n = \frac{Z_k}{k^2} \left(1 \pm O(k/n) \pm O(1/k)\right).$$

Therefore, recalling notation of a_n, we have that

$$|a_m - a_n| = a_k O\left(\frac{k}{\min\{m,n\}}\right) + a_k O(1/k).$$

Since, $a_k \in [1, \alpha]$ for all k, we obtain the desired result of Lemma 13. ■

IX. Conclusion

In this paper, we present simple novel local approximation algorithm for computing log-partition function and MAP estimation for arbitrary exponential distribution represented by a pair-wise MRF. We showed these algorithms provide bounds for arbitrary graph with quantifiable approximation guarantees. Further, for low-doubling dimension graphs and minor-excluded graphs it can provide arbitrary accuracy within linear time. The main takeaway for a practitioner is the following: there is a simple and intuitive local algorithm that provides provable bounds with computable approximation error for any graph and hence it can be used as a good heuristic and producing approximation guarantee certificate.

We proposed message-passing implementation based on self-avoiding walk trees which should provide such implementation for other problems as well. This method, through a transformation from non-binary exponential family to binary MRF, extends for any finite valued factor graph. However, this can result in somewhat redundant construction. Understanding design of direct constructions for non-binary pair-wise MRF is an important open problem.

We derived an unusual implication of our algorithmic results for providing existence of asymptotic limits of free energy for a class of regular MRFs. Our result suggest a way to explicitly evaluate these limiting up to an arbitrary accuracy. This should be of general interest as a method for establishing asymptotic limits as well as computing these limits.

Finally, we remark that our methods are explained for exponential family only. However, they easily extend to certain hard-core models such as independent set or matching where there is a *non-constraining* assignment to node values.

References

[1] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. San Francisco, CA: Morgan Kaufmann, 1988.

[2] M. Wainwright and M. Jordan, “Graphical models, exponential families, and variational inference,” *UC Berkeley, Dept. of Statistics, Technical Report 649*, 2003.

[3] A. Sinclair and M. Jerrum, “Approximate counting, uniform generation and rapidly mixing markov chains,” *Inf. Comput.*, vol. 82, no. 1, pp. 93–133, 1989.

[4] F. P. Kelly, “Loss networks,” *Annals of Applied Probability*, vol. 1, pp. 319–378, 1991.

[5] J. Yedidia, W. Freeman, and Y. Weiss, “Generalized belief propagation,” *Mitsubishi Elect. Res. Lab., TR-2000-26*, 2000.

[6] M. J. Wainwright, T. Jaakkola, and A. S. Willsky, “Tree-based reparameterization framework for analysis of sum-product and related algorithms,” *IEEE Transactions on Information Theory*, 2003.

[7] ——, “Map estimation via agreement on (hyper)trees: Message-passing and linear-programming approaches,” *IEEE Transactions on Information Theory*, 2005.

[8] ——, “A new class of upper bounds on the log partition function,” *IEEE Transactions on Information Theory*, 2005.

[9] S. C. Tatikonda and M. I. Jordan, “Loopy belief propagation and gibbs measure,” in *Uncertainty in Artificial Intelligence*, 2002.

[10] M. Bayati, D. Shah, and M. Sharma, “Max-product for maximum weight matching: convergence, correctness and lp duality,” *IEEE Transaction on Information Theory*, Accepted, preliminary version appeared in IEEE, ISIT 2005.

[11] C. Moallemi and B. V. Roy, “Convergence of the min-sum message passing algorithm for quadratic optimization,” *Stanford University Technical report*, 2006.

[12] V. Kolmogorov and M. Wainwright, “On optimality of tree-reweighted max-product message-passing,” in *Uncertainty in Artificial Intelligence*, 2005.

[13] V. Kolmogorov, “Convergent tree-reweighted message passing for energy minimization,” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2006.

[14] A. Globerson and T. Jaakkola, “Bound on partition function through planar graph decomposition,” in *NIPS*, 2006.

[15] P. Klein, S. Plotkin, and S. Rao, “Excluded minors, network decomposition and multicommodity flow,” in *ACM STOC*, 1993.

[16] S. Rao, “Small distortion and volume preserving embeddings for planar and euclidean metrics,” in *SCG ’99: Proceedings of the fifteenth annual symposium on Computational geometry*. New York, NY, USA: ACM Press, 1999, pp. 300–306.

[17] N. Madras and G. Slade, *The Self-Avoiding Walk*. Birkhauser, Boston, 1993.

[18] D. Weitz, “Counting independent sets up to the tree threshold,” in *STOC ’06: Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*. New York, NY, USA: ACM Press, 2006, pp. 140–149.

[19] D. R. Karger and M. Ruhl, “Finding nearest neighbors in growth-restricted metrics,” in *STOC ’02: Proceedings of the thiry-fourth annual ACM symposium on Theory of computing*. New York, NY, USA: ACM Press, 2002, pp. 741–750.

[20] A. Gupta, R. Krauthgamer, and J. R. Lee, “Bounded geometries, fractals, and low-distortion embeddings,” in *FOCS ’03: Proceedings of the 44th Annual IEEE Symposium on Foundations of Computer Science*. Washington, DC, USA: IEEE Computer Society, 2003, p. 534.

[21] S. Har-Peled and S. Mazumdar, “On coresets for k-means and k-median clustering,” in *STOC ’04: Proceedings of the thirty-sixth annual ACM symposium on Theory of computing*. New York, NY, USA: ACM Press, 2004, pp. 291–300.

[22] N. Robertson and P. Seymour, “The graph minor theory,” started, 1984.

[23] S. Sanghavi, D. Shah, and A. Willsky, “Max-product for maximum weight independent set,” *Pre-print*, 2007.

APPENDIX A
PROOF OF LEMMA 1

The proof is by induction on $r \in \mathbb{N}$. For base case, consider $r = 0$. Now, $\mathbf{B}(x, 2^0 = 1)$ is essentially the set of all points which are at distance < 1 by definition. Since it is metric with distance being integer, this means that the set of all points that are at distance 0. By definition of metric, we have that x is the only such point. That is, $\mathbf{B}(x, 1) = \{x\}$. Hence, $|\mathbf{B}(x, 1)| = 1 \leq 2^{0\times\rho(\mathcal{M})}$ for all $x \in \mathcal{X}$.

Now suppose the claim of Lemma is true for all $r \leq k$ and all $x \in \mathcal{X}$. Consider $r = k + 1$ and any $x \in \mathcal{X}$. By definition of doubling dimension, there exists $\ell \leq 2^{\rho(\mathcal{M})}$ balls of radius 2^k, say $\mathbf{B}(y_j, 2^k)$ with $y_j \in \mathcal{X}$ for $1 \leq j \leq \ell$, such that

$$\mathbf{B}(x, 2^{k+1}) \subset \cup_{j=1}^{\ell} \mathbf{B}(y_j, 2^k).$$

Therefore,

$$|\mathbf{B}(x, 2^{k+1})| \leq \sum_{j=1}^{\ell} |\mathbf{B}(y_j, 2^k)|.$$

By inductive hypothesis, for $1 \leq j \leq \ell$,

$$|\mathbf{B}(y_j, 2^k)| \leq 2^{k\rho(\mathcal{M})}.$$

Since we have $\ell \leq 2^{\rho(\mathcal{M})}$, we obtain

$$|\mathbf{B}(x, 2^{k+1})| \leq \ell\ 2^{k\rho(\mathcal{M})} \ \leq\ 2^{(k+1)\rho(\mathcal{M})}.$$

This completes the proof of inductive step and that of the Lemma 1.

APPENDIX B
TRANSFORMATION: MAP IN FACTOR GRAPH TO BINARY PAIR-WISE MRF

In this section we show that any MAP estimation problem is equivalent to estimating MAP in a specific binary pair-wise problem on a suitably constructed graph with node potentials. This construction is from work by Sanghavi, Shah and Willsky [23]. This construction is related to the "overcomplete basis" representation [2]. Consider the following canonical MAP estimation problem: suppose we are given a distribution $q(\mathbf{y})$ over vectors $\mathbf{y} = (y_1, \ldots, y_M)$ of variables y_m, each of which can take a finite value. Suppose also that q factors into a product of strictly positive functions, which we find convenient to denote in exponential form:

$$q(\mathbf{y}) \ = \ \frac{1}{Z}\prod_{\alpha\in A}\exp\left(\phi_\alpha(\mathbf{y}_\alpha)\right) \ = \ \frac{1}{Z}\exp\left(\sum_{\alpha\in A}\phi_\alpha(\mathbf{y}_\alpha)\right)$$

Here α specifies the domain of the function ϕ_α, and $\mathbf{y}_\alpha$ is the vector of those variables that are in the domain of ϕ_α. The α's also serve as an index for the functions. A is the set of functions. The MAP estimation problem is to find a maximizing assignment $\mathbf{y}^* \in \arg\max_{\mathbf{y}} q(\mathbf{y})$.

We now build an auxillary graph $\widetilde{G}$, and assign weights to its nodes, such that the MAP estimation problem above is equivalent to finding the MWIS of $\widetilde{G}$. There is one node in $\widetilde{G}$ for each pair $(\alpha, \mathbf{y}_\alpha)$, where $\mathbf{y}_\alpha$ is an *assignment* (i.e. a set of values for the variables) of domain α. We will denote this node of $\widetilde{G}$ by $\delta(\alpha, \mathbf{y}_\alpha)$.

There is an edge in $\widetilde{G}$ between any two nodes $\delta(\alpha_1, \mathbf{y}^1_{\alpha_1})$ and $\delta(\alpha_2, \mathbf{y}^2_{\alpha_2})$ if and only if there exists a variable index m such that

1) m is in both domains, i.e. $m \in \alpha_1$ and $m \in \alpha_2$, and
2) the corresponding variable assignments are different, i.e. $y^1_m \neq y^2_m$.

In other words, we put an edge between all pairs of nodes that correspond to *inconsistent* assignments. Given this graph $\widetilde{G}$, we now assign weights to the nodes. Let $c > 0$ be any number such that $c + \phi_\alpha(\mathbf{y}_\alpha) > 0$ for all α and $\mathbf{y}_\alpha$. The existence of such a c follows from the fact that the set of assignments and domains is finite. Assign to each node $\delta(\alpha, \mathbf{y}_\alpha)$ a weight of $c + \phi_\alpha(\mathbf{y}_\alpha)$. Consider an example of this construction first. Later, we state the precise equivalence.

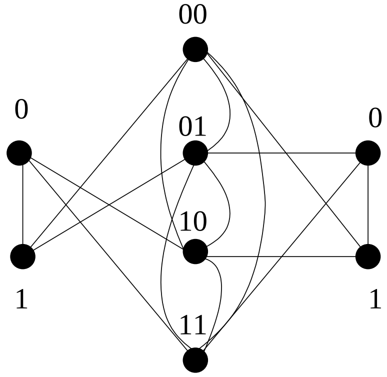

Fig. 10. Example of transforming MAP for factor graph to MAP in binary pair-wise MRF.

Example 2: Let y_1 and y_2 be binary variables with joint distribution

$$q(y_1, y_2) \ = \ \frac{1}{Z}\exp(\theta_1 y_1 + \theta_2 y_2 + \theta_{12} y_1 y_2)$$

where the θ are any real numbers. The corresponding $\widetilde{G}$ is shown in Figure 10. Let c be any number such that $c + \theta_1$, $c + \theta_2$ and $c + \theta_{12}$ are all greater than 0. The weights on the nodes in $\widetilde{G}$ are: $\theta_1 + c$ on node "1" on the left, $\theta_2 + c$ for node "1" on the right, $\theta_{12} + c$ for the node "11", and c for all the other nodes.

Lemma 14: Suppose q and $\widetilde{G}$ are as above. (a) If $\mathbf{y}^*$ is a MAP estimate of q, let $\delta^* = \{\delta(\alpha, \mathbf{y}^*_\alpha) \,|\, \alpha \in A\}$ be the set of nodes in $\widetilde{G}$ that correspond to each domain being consistent with $\mathbf{y}^*$. Then, δ^* is an MWIS of $\widetilde{G}$. (b) Conversely, suppose δ^* is an MWIS of $\widetilde{G}$. Then, for every domain α, there is exactly one node $\delta(\alpha, \mathbf{y}^*_\alpha)$ included in δ^*. Further, the corresponding domain assignments$\{\mathbf{y}^*_\alpha \,|\, \alpha \in A\}$ are consistent, and the resulting overall vector $\mathbf{y}^*$ is a MAP estimate of q.

Proof: A *maximal* independent set is one in which every node is either in the set, or is adjacent to another node that is in the set. Since weights are positive, any MWIS has to be maximal. For $\widetilde{G}$ and q as constructed, it is clear that

1) If $\mathbf{y}$ is an assignment of variables, consider the corresponding set of nodes $\{\delta(\alpha, \mathbf{y}_\alpha) \,|\, \alpha \in A\}$. Each domain α has exactly one node in this set. Also, this set is an independent set in $\widetilde{G}$, because the partial assignments

$\mathbf{y}_\alpha$ for all the nodes are consistent with $\mathbf{y}$, and hence with each other. This means that there will not be an edge in $\widetilde{G}$ between any two nodes in the set.

2) Conversely, if Δ is a maximal independent set in $\widetilde{G}$, then all the sets of partial assignments corresponding to each node in Δ are all consistent with each other, and with a global assignment $\mathbf{y}$.

There is thus a one-to-one correspondence between maximal independent sets in $\widetilde{G}$ and assignments $\mathbf{y}$. The lemma follows from this observation. ■