Max Product for Max-Weight Independent Set and Matching

Devavrat Shah
MIT
Email: devavrat@mit.edu

Abstract

The Max Product (MP) is a local, iterative, message passing style algorithm that has been developed for finding the maximum a posteriori (MAP) assignment of discrete probability distribution specified by a graphical model. The scope of application of MP is vast and in particular it can serve as a heuristic to solve any combinatorial optimization problem. Despite the success of MP algorithm in the context of coding and vision, not much has been theoretically understood about the correctness and convergence of MP.

The Maximum Weight Independent Set (MWIS) and Maximum Weight Matching (MWM) are classically well studied combinatorial optimization problems. A lot of work has been done to design efficient algorithms for finding MWIS and MWM. In this paper, we study application of MP algorithm for MWIS and MWM for sparse random graphs: $G(n, c/n)$ and $G_r(n)$, which are n node random graphs with parameter c and r respectively. We show that when weights (node or edge depending on MWIS or MWM) are assigned independently according to exponential distribution, the MP algorithm converges and finds correct solution for a large range of parametric value c and r. In particular, we show that for any $\epsilon > 0$, for large enough n, the MP becomes $1 + \epsilon$ competitive with probability at least $1 - \epsilon$.

Our results build upon the results of Gamarnik, Nowicki and Swirscsz (2005), which established *local optimality property* of MWIS and MWM for sparse random graphs.

1 Introduction

Graphical models (GM) are a powerful method for representing and manipulating joint probability distributions. They have found major applications in several different research communities such as artificial intelligence [16], statistics [13], error-control coding [11] and neural networks. Two central problems in probabilistic inference over graphical models are those of evaluating the *marginal* and *maximum a posteriori* (MAP) probabilities, respectively. In general, calculating the marginal or MAP probabilities for an ensemble of random variables would require a complete specification of the joint probability distribution. Further, the complexity of a brute force calculation would be exponential in the size of the ensemble. GMs assist in exploiting the dependency structure between the random variables, allowing for the design of efficient inference algorithms.

The belief propagation (BP) and max-product algorithms [16] were proposed in order to compute, respectively, the marginal and MAP probabilities efficiently. Comprehensive surveys of various formulations of BP and its generalization, the junction tree algorithm, can be found in [1, 25, 19]. BP-based message-passing algorithms have been very successful in the context of, for example, iterative decoding for turbo codes and in computer vision. The simplicity, wide scope of application and experimental success of belief propagation has attracted a lot of attention recently [1, 12, 17, 24].

BP is known to converge to the correct marginal/MAP probabilities on tree graphs [16] or graphs with a single loop [2, 21]. For graphical models with arbitrary underlying graphs, little is known about the correctness of BP. Partial progress consists of [22] where correctness of BP for Gaussian GMs is proved,

[10] where an attenuated modification of BP is shown to work, and [17] where the iterative turbo decoding algorithm based on BP is shown to work in the asymptotic regime with probabilistic guarantees. To the best of our knowledge, little theoretical progress has been in resolving the question: Why does BP work on arbitrary graphs?

Motivated by the objective of providing justification for the success of BP on arbitrary graphs, we focus on the application of BP to the two well-known combinatorial optimization problems: Finding (1) Maximum Weight Independent Set (MWIS) and (2) Maximum Weight Matching (MWM), in an arbitrary graph. It is standard to represent combinatorial optimization problems, like finding the MWIS and MWM, as calculating the MAP probability on a suitably defined GM which encodes the data and constraints of the optimization problem. Thus, the max-product algorithm can be viewed at least as a heuristic for solving the problem. In this paper, we study the performance of the max-product algorithm as a method for finding the MWIS and MWM on a weighted graph.

It has been empirically observed that MP algorithm works well on random instances of hard optimization problem. It has been widely believed that the "large girth" property of such random instances is responsible for this success. The main result of this paper provides justification of this observation in the context of MWIS and MWM. In particular, we show that the MP algorithm converges to correct MWIS or MWM when (1) the underlying graph has large girth, that is, if lengths of all cycles in the graph are very large (defined precisely later in the paper), and (2) the weight (node or edge) are assigned independently according to appropriate distribution. Next, we describe setup, related work and main results.

1.1 Setup

Graph. Consider an undirected graph $G = (V, E)$ with vertex set V and edge-set E. Let the number of nodes be n, i.e. $|V| = n$, and $(i, j) \in E$ iff nodes i and j are connected to each other. Let $d(G)$ denote the length of the shortest cycle in the graph G. To each node and edge, non-negative real valued weight is assigned. Let w_i denote weight of node i and w_{ij} denote weight of edge $(i, j) \in E$. In this paper, we consider sparse random graphs, $G(n, c/n)$ and $G_r(n)$ described as follows:

1. The $G(n, c/n)$ has n nodes. An edge is present between any node-pair i, j with probability c/n independently.

2. The $G_r(n)$ has n nodes. It is formed by sampling one of the r-regular n node graph uniformly at random.

It is well known that both $G(n, c/n)$ and $G_r(n)$ have *large* $d(G)$ for any constant c, r with high probability. In particular, $d(G) \to \infty$ as $n \to \infty$ with high probability. The weights (both node and edge) are assigned in an i.i.d. fashion according to a certain distribution. Our interest will be in the exponential distribution of mean 1. In this paper, we follow the notation that the random variables denoting weights (node or edge) will be represented in capital letters (such as W_i) while specific instance will be represented in small letters (such as w_i).

Independent set. A subset of nodes, say $\mathcal{I} \subset V$, is called independent set if no two nodes $u, v \in \mathcal{I}$ have edge between them. Weight of an independent set $\mathcal{I}$, denoted by $w(\mathcal{I})$, is the sum of the weights of node in $\mathcal{I}$, that is

$$w(\mathcal{I}) = \sum_{j \in \mathcal{I}} w_j.$$

Let $\mathcal{I}^*$ denote a maximum weighted independent set (MWIS), that is

$$\mathcal{I}^* = \arg\max_{\mathcal{I}} w(\mathcal{I}).$$

In presence of multiple MWIS, let $\mathcal{I}^*$ be any one of them chosen arbitrarily.

Matching. A subset of edges, say $\mathcal{M} \subset E$, is called matching if no two edges of $\mathcal{M}$ share a vertex. Weight of a matching $\mathcal{M}$, denoted by $w(\mathcal{M})$, is the sum of the weights of edges in $\mathcal{M}$, that is

$$w(\mathcal{M}) = \sum_{(i,j)\in\mathcal{M}} w_{ij}.$$

Let $\mathcal{M}^*$ denote the maximum weight matching (MWM), that is

$$\mathcal{M}^* = \arg\max_{\mathcal{M}} w(\mathcal{M}).$$

In presence of multiple MWM, let $\mathcal{M}^*$ be any one of them chosen arbitrarily.

1.2 Related work

In this section, we briefly describe previous work related to MWIS and MWM. Both MWIS and MWM are well studied combinatorial optimization problems. Hence, it is difficult to be comprehensive in reporting all relevant work.

First, Maximum Weight Matching. The other variant of MWM is the Minimum Weight Matching, known as the assignment problem. Both MWM and the assignment problems are algorithmically equivalent. Attempts to find better MWM algorithms contributed to the development of the rich theory of network flow algorithms [8, 14]. The assignment problem has been studied in various contexts such as job-assignment in manufacturing systems [8], switch scheduling algorithms [15] and auction algorithms [6]. Recently, Bayati et. al. (2005) [31] showed that MP finds MWM in a complete bipartite graph for arbitrary weight as long as it is unique. They show that MP has complexity similar to that of Auction algorithm or Edmond-Karp's algorithm for integer weights. Unfortunately, their results do not extend for arbitrary graphs. In particular, their results do not say anything about the performance of MP for sparse random graphs.

Next, Maximum Weight Independent Set. Unlike MWM, the MWIS is known to be NP-hard and hard to approximate within constant factor. From both algorithm design and complexity perspective, this problem has been very well studied. Many different algorithmic approaches have been designed to find good algorithms.

Now, on MWIS and MWM for random graphs with random weights. A lot of work since early 1980s has concentrated on evaluating asymptotic value of combinatorial optimization problem under natural probabilistic setting. Among the first such results was due to Karp and Sipser[32]. They showed that for sparse random graph, $G(n, c/n)$ for $c \leq e$, a simple linear time algorithm finds maximum independent set and maximum size matching. As a consequence of this, they obtained exact asymptotic formula for the size of maximum size matching ! Subsequently, there have been many results on evaluating exact asymptotic answers for combinatorial optimization problems in probabilistic setup. An excellent reference for such results is monograph by Steele [34]. In his seminal work, Aldous [4, 3] and Aldous and Steele [26] proposed method of local weak convergence (LWC) to establish existence of asymptotic limits for combinatorial optimization problem. such as the assignment problem. A recent survey of Aldous and Bandopadhyay [5] presents nice frame-work for evaluating such asymptotic limits as a solution of recursive equations. Recently, the remarkable results of Gamarnik [33] and Gamarnik, Nowicki and Swirscsz [27] build upon the LWC method to establish asymptotic limits for combinatorial optimization problems such as random linear constraint satisfaction problem, MWIS and MWM. In summary, the above results have established existence of asymptotic limits for optimization problems and provided means to evaluate them. Additionally, these result establish the following *local optimality property* – decision related to a node under optimization problem depends on its *local* neighborhood. Thus, these results suggest that these problems should become easy asymptotically. Further, these provide hope for algorithms like Max-Product to be effective in such setup.

However, these results do not imply anything about the convergence or correctness of Max Product (Belief Propagation) algorithms.

Finally, on convergence results for Max Product or Belief Propagation. Initial proposal of Belief propagation or Max Product was meant to work on trees [**?**]. Earlier in Section 1, we have stated the evolution and success of BP or MP algorithm.

1.3 Graphical model: MWIS and MWM

Next, we model the problem of finding MWIS and MWM as finding a MAP assignment in a graphical model where the joint probability distribution can be completely specified in terms of the product of functions that depend on at most two variables (nodes). For details about GMs, we urge the reader to see [13].

GM for MWIS. Now, consider the following GM^{IS} defined on G: Let $X_1, \ldots, X_n$ be random variables corresponding to the vertices of G and taking values in $\{0, 1\}$. The $X_i = 1$ corresponds to node i being present in a set and $X_i = 0$ corresponds to absence. Next, we define a joint probability distribution, denoted by P^{IS}, on $\overline{X} \in \{0, 1\}^n$ as follows:

$$P^{IS}\left(\overline{X}\right) \;=\; \frac{1}{Z} \prod_{(i,j)\in E} \psi_{ij}^{IS}(x_i, x_j) \prod_i \phi_i^{IS}(x_i), \tag{1}$$

where Z is the normalization constant, the compatibility functions, $\psi_{\cdot\cdot}^{IS}(\cdot,\cdot)$, are defined as

$$\psi_{ij}^{IS}(r, s) = \begin{cases} 0 & \text{if } r = 1 \text{ and } s = 1 \\ 1 & \text{Otherwise} \end{cases}$$

and the potentials at the nodes, $\phi_{\cdot}^{IS}(\cdot)$, are defined as

$$\phi_i(r) = \begin{cases} e^{w_i} & \text{if } r = 1 \\ 1 & \text{Otherwise} \end{cases}$$

The above defined GM^{IS} is also called pair-wise Markov random field. The following claims are a direct consequence of these definitions.

Claim 1.1. *For the GM^{IS} as defined above, the joint density $P^{IS}\left(\overline{X} = (x_1, \ldots, x_n)\right)$ for $\overline{X} \in \{0, 1\}^n$ is nonzero if and only if the subset $\mathcal{I}(\overline{X}) = \{i \in V : x_i = 1\}$ is an independent set. Further, when nonzero*

$$P^{IS}\left(\overline{X}\right) = \frac{1}{Z} e^{w(\mathcal{I}(\overline{X}))}.$$

Claim 1.2. *Let $\overline{X}^* \in \{0, 1\}^n$ be such that*

$$\overline{X}^* = \arg\max\{P^{IS}\left(\overline{X}\right)\}.$$

Then, the corresponding $\mathcal{I}(\overline{X}^*)$ is an MWIS in G.

GM for MWM. Now, consider the following GM^{M} defined on G: Let $X_1, \ldots, X_n$ be random variables corresponding to the vertices of G such that $X_i \in \{\star\} \cup \mathcal{N}(i)$, where $\mathcal{N}(i)$ is the set of neighbors of i, i.e.

$$\mathcal{N}(i) = \{j \in V : (i, j) \in E\}.$$

The $X_i = j \in \mathcal{N}(i)$ corresponds to node i being connected to node j while $X_i = \star$ corresponds to node i not connected to any other node. Next, we define a joint probability distribution, denoted by P^M, on $\overline{X} = (x_1, \ldots, x_n)$ such that $x_i \in \{\star\} \cup \mathcal{N}(i)$, as follows:

$$P^M\left(\overline{X}\right) = \frac{1}{Z} \prod_{(i,j)\in E} \psi_{ij}^M(x_i, x_j) \prod_i \phi_i^M(x_i), \tag{2}$$

where Z is the normalization constant, the compatibility functions, $\psi_{\cdot\cdot}^M(\cdot,\cdot)$, are defined as

$$\psi_{ij}^{IS}(r,s) = \begin{cases} 0 & \text{if } r = j \text{ and } j \neq i \\ 0 & \text{if } s = i \text{ and } r \neq j \\ 1 & \text{Otherwise} \end{cases}$$

and the potentials at the nodes, $\phi_{\cdot}^M(\cdot)$, are defined as

$$\phi_i(r) = \begin{cases} e^{w_{ir}} & \text{if } r \in \mathcal{N}(i) \\ 1 & \text{Otherwise} \end{cases}$$

The above defined GM^M is also called pair-wise Markov random field. The following claims are a direct consequence of these definitions.

Claim 1.3. *For the GM^M as defined above, the joint density $P^M\left(\overline{X} = (x_1, \ldots, x_n)\right)$ for $x_i \in \{\star\} \cup \mathcal{N}(i)$ is nonzero if and only if the subset $\mathcal{M}(\overline{X}) = \{(i,j) \in E : x_i = j\}$ is a matching. Further, when nonzero*

$$P^M\left(\overline{X}\right) = \frac{1}{Z} e^{2w(\mathcal{M}(\overline{X}))}.$$

Claim 1.4. *Let $\overline{X}^*$ be such that*

$$\overline{X}^* = \arg\max\{P^M\left(\overline{X}\right)\}.$$

Then, the corresponding $\mathcal{M}(\overline{X}^*)$ is an MWM in G.

1.4 Max-Product and Min-Sum Algorithms

The claims 1.2 and 1.4 imply that finding the MWIS and MWM respectively are equivalent to finding the maximum a posteriori (MAP) assignment on the GM^{IS} and GM^M respectively. Thus, the standard max-product algorithm can be used as an iterative strategy for finding MWIS and MWM. Before we describe the max-product and equivalent min-sum algorithm algorithm for MWIS and MWM, we need some definitions.

Definition 1. *For any $p \in \mathbb{N}$, let $D \in \mathbb{R}^{p\times p}$ be any $p \times p$ matrix and $X, Y, Z \in \mathbb{R}^{p\times 1}$. Then the operations $*, \odot$ are defined as follows:*

$$D * X = Z \iff z_i = \max_{1\leq j\leq p} d_{ij}x_j, \ \forall 1 \leq i \leq p, \tag{3}$$

$$X \odot Y = Z \iff z_i = x_i y_i, \ \forall 1 \leq i \leq p. \tag{4}$$

For $X_1, \ldots, X_m \in \mathbb{R}^{p\times 1}$,

$$\bigodot_{i=1}^{m} X_i = X_1 \odot X_2 \odot \ldots \odot X_m. \tag{5}$$

Max-Product for MWIS.

For $(i,j) \in E$, define a 2×2 compatibility matrix $\Psi_{ij}^{IS} \in \mathbb{R}^{2\times 2}$ such that its (r,s) entry is $\psi_{ij}^{IS}(r,s)$, for $1 \le i,j \le n$. Also, let $\Phi_i^{IS} \in \mathbb{R}^{2\times 1}$ be the following:

$$\Phi_i^{IS} = [\phi_{\alpha_i}^{IS}(0), \phi^{IS}(1)]^T.$$

In the description of algorithm, we will not indicate the super-script IS (and later M) as it is clear from context that the Ψ and Φ correspond to IS (or M).

Max-Product Algorithm (MWIS).

---

(1) Let $M_{i\to j}^k = [m_{i\to j}^k(0), m_{i\to j}^k(1)]^T \in \mathbb{R}^{2\times 1}$ denote the messages passed from i to j in the iteration $k \ge 0$, for every $(i,j) \in E$ (or $(j,i) \in E$).

(2) Initially $k = 0$ and set the messages as follows. Let $M_{i\to j}^0 = [m_{i\to j}^0(0), m_{i\to j}^0(1)]^T$ where

$$m_{i\to j}^0(r) = \begin{cases} e^{w_i} & \text{if } r = 0 \\ 1 & \text{otherwise} \end{cases} \tag{6}$$

(3) For $k \ge 1$, messages in iteration k are obtained from messages of iteration $k-1$ recursively as follows: for every $(i,j) \in E$,

$$M_{i\to j}^k \;=\; \Psi_{ij}^T * \left\{ \left(\bigodot_{\ell \in \mathcal{N}(i)\setminus\{j\}} M_{\ell\to i}^{k-1} \right) \odot \Phi_i \right\} \tag{7}$$

(4) Define the beliefs (2×1 vectors) at each node $i \in V$, in iteration k as follows.

$$b_i^k \;=\; \left(\bigodot_{\ell \in \mathcal{N}(i)} M_{\ell\to i}^k \right) \odot \Phi_i. \tag{8}$$

(5) The estimated MWIS at the end of iteration k is $\mathcal{I}^k$, represented by $\overline{X}^k = (x_1^k, \ldots, x_n^k)$ where

$$x_i^k = \mathbf{1}_{\{b_i^k(1) > b_i^k(0)\}}.$$

(6) Repeat (3)-(5) till $\mathcal{I}^k$ converges.

Note 2. For computational stability, it is often recommended that messages be normalized at every iteration. However, such normalization does not change the output of the algorithm. Since we are only interested in theoretically analyzing the algorithm, we will ignore the normalization step. Also, the messages are usually all initialized to one. Although the result doesn't depend on the initial values, setting them as defined above makes the analysis and formulas nicer at the end.

Min-Sum for MWIS.

The max-product and min-sum algorithms can be seen to be equivalent by observing that the logarithm function is monotone and hence $\max_i \log(\alpha_i) = \log(\max_i \alpha_i)$. In order to describe the min-sum algorithm, we need to redefine $\Phi_i,\ 1 \leq i \leq n$, as follows:

$$\Phi_i = [0, w_i]^T.$$

Now, the min-sum algorithm is exactly the same as max-product with the equations (6), (7) and (11) replaced by the following equations respectively.

(a) Replace (6) by the following.

$$m^0_{i\to j}(r) = \begin{cases} w_i & \text{if } r = 0 \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

(b) Replace (7) by the following.

$$M^k_{i\to j} \quad = \quad \Psi^T_{ij} * \left\{ \left(\sum_{\ell \in \mathcal{N}(i)\setminus\{j\}} M^{k-1}_{\ell\to i} \right) + \Phi_i \right\} \tag{10}$$

(c) Replace (11) by the following.

$$b^k_i \quad = \quad \left(\sum_{\ell \in \mathcal{N}(i)} M^k_{\ell\to i} \right) + \Phi_i. \tag{11}$$

Note 3. The min-sum algorithm involves only summations and subtractions compared to max-product which involves multiplications and divisions. Computationally, this makes the min-sum algorithm more efficient and hence very attractive.

Min-Sum for MWM.

We describe only Min-Sum algorithm for MWM as this paper will analyze Min-Sum algorithm. The difference between Min-Sum for MWIS and MWM is mainly in the compatibility matrix $\Psi_{..}$ and potential matrix $\Phi_{..}$. With abuse of notation, re-define the compatibility matrix as follows: for $(i, j) \in E$, define a $n + 1 \times n + 1$ compatibility matrix $\Psi_{ij} \in \mathbb{R}^{n+1\times n+1}$ such that

$$\Psi_{ij}(r, s) = \begin{cases} 0 & \text{if } r = j \text{ and } s \neq i \\ 0 & \text{if } s = i \text{ and } r \neq j \\ 1 & \text{otherwise} \end{cases}$$

Also, let $\Phi_i \in \mathbb{R}^{n+1\times 1}$ be such that

$$\Phi_i(j) = \begin{cases} w_{ij} & \text{if } (i, j) \in E \\ 0 & \text{otherwise} \end{cases}$$

Min-Sum Algorithm (MWM).

(1) Let $M^k_{i\to j} = [m^k_{i\to j}(1), \ldots, m^k_{i\to j}(n+1)]^T \in \mathbb{R}^{n+1\times 1}$ denote the messages passed from i to j in the iteration $k \geq 0$, for every $(i,j) \in E$ (or $(j,i) \in E$).

(2) Initially $k = 0$ and set the messages as follows. Let $M^0_{i\to j} = [m^0_{i\to j}(1), \ldots, m^0_{i\to j}(n+1)]^T$ where

$$m^0_{i\to j}(r) = \begin{cases} w_{ij} & \text{if } r = j \\ 0 & \text{otherwise} \end{cases} \tag{12}$$

(3) For $k \geq 1$, messages in iteration k are obtained from messages of iteration $k-1$ recursively as follows: for every $(i,j) \in E$,

$$M^k_{i\to j} \quad = \quad \Psi^T_{ij} * \left\{ \left(\sum_{\ell \in \mathcal{N}(i)\setminus\{j\}} M^{k-1}_{\ell\to i} \right) + \Phi_i \right\} \tag{13}$$

(4) Define the beliefs ($n+1\times 1$ vectors) at each node $i \in V$, in iteration k as follows.

$$b^k_i \quad = \quad \left(\sum_{\ell \in \mathcal{N}(i)} M^k_{\ell\to i} \right) + \Phi_i. \tag{14}$$

(5) The estimated MWM at the end of iteration k is $\mathcal{M}^k$, represented by $\overline{X}^k = (x^k_1, \ldots, x^k_n)$ where

$$x^k_i = \arg \max_{j \in \mathcal{N}(i)\cup\{n+1\}} \{b^k_i(j)\}.$$

We note that $x^k_i = n+1$ means that node i is not connected to any other node in that matching.

(6) Repeat (3)-(5) till $\mathcal{M}^k$ converges.

1.5 Main Result

1.5.1 Result for Min-Sum Algorithm for MWIS

We first state a little modification of min-sum algorithm for MWIS before stating the result. This modification is related to stopping condition.

Modification. Stop algorithm after large enough k. Consider the decisions (x^k_i) at the end of iteration k. The subset, $\mathcal{I}^k$ induced by (x^k_i) may not be independent set. We state a simple iterative procedure (can be made local) to obtain an independent set out of $\mathcal{I}^k$. Intially, set $\hat{\mathcal{I}}^k = \mathcal{I}^k$. Consider nodes in $\hat{\mathcal{I}}^k$ in any order and repeat the following till possible: if a node i is inside $\hat{\mathcal{I}}^k$ and one or more of its neighbors are also in $\hat{\mathcal{I}}^k$, remove i and its neighbors from $\hat{\mathcal{I}}^k$. By definition, at the end the $\hat{\mathcal{I}}^k$ is an independent set.

Theorem 1.1. *Consider graph $G(n, c/n)$ or $G_r(n)$ with node weights assigned independently according to exponential distribution of rate 1. Let $c \leq 2e$ and $r \leq 4$. Then, for any $\epsilon > 0$, there exists large enough $N^{IS}(\epsilon)$ and $T^{IS}(\epsilon)$ such that if $n > N^{IS}(\epsilon)$, then the following holds:*

(a) For any node in $G(n, c/n)$ or $G_r(n)$, say i, the x_i^k converges with probabilty[1] *at least $1-\epsilon$ for $k \geq T^{IS}(\epsilon)$.*

(b) Let $\hat{\mathcal{I}}^k$ be the independent set obtained by modifying the set $\mathcal{I}^k$ obtained at the end of iteration k of min-sum algorithm. Then, the weight of $\hat{\mathcal{I}}^k$, $W(\hat{\mathcal{I}}^k)$ is such that for $k \geq T^{IS}(\epsilon)$,

$$P\left(\frac{|W(\hat{\mathcal{I}}^k) - W(\mathcal{I}^*)|}{W(\mathcal{I}^*)} \geq \delta(\epsilon)\right) \leq \epsilon,$$

where $\delta(\epsilon) \to 0$ as $\epsilon \to 0$.

1.5.2 Result for Min-Sum Algorithm for MWM.

Similar to MWIS, we make the following modification to stopping condition of min-sum algorithm algorithm for MWM.

Modification. Stop algorithm after large enough k. Consider the decisions (x_i^k) at the end of iteration k. The subset of edges, $\mathcal{M}^k$ induced by (x_i^k) may not be a matching. We state a simple iterative procedure (can be made local) to obtain a matching out of $\mathcal{M}^k$. Intially, set $\hat{\mathcal{M}}^k = \mathcal{M}^k$. Consider edges in $\hat{\mathcal{M}}^k$ in any order and repeat the following till possible: if an edge $(i,j) \in \hat{\mathcal{M}}^k$ shares an end-point with any edge in $\hat{\mathcal{M}}^k$, remove (i,j) and the conflicting edge from $\hat{\mathcal{M}}^k$. By definition, at the end the $\hat{\mathcal{M}}^k$ is a matching.

Theorem 1.2. *Consider graph $G(n, c/n)$ or $G_r(n)$ with edge weights assigned independently according to exponential distribution of rate 1. Let $c > 0$ and $r \geq 1$. Then, for any $\epsilon > 0$, there exists large enough $N^M(\epsilon), T^M(\epsilon)$ such that for $n > N^M(\epsilon)$ the following holds:*

(a) For any node in $G(n, c/n)$ or $G_r(n)$, say i, the x_i^k converges with probabilty[2] *at least $1-\epsilon$ for $k \geq T^M(\epsilon)$.*

(b) Let $\hat{\mathcal{M}}^k$ be the matching obtained by the modifying $\mathcal{M}^k$ obtained at the end of iteration k of the min-sum algorithm. Then, the weight of $\hat{\mathcal{M}}^k$, $W(\mathcal{M}^k)$ is such that for $k \geq T^M(\epsilon)$,

$$P\left(\frac{|W(\hat{\mathcal{M}}^k) - W(\mathcal{M}^*)|}{W(\mathcal{M}^*)} \geq \delta(\epsilon)\right) \leq \epsilon,$$

where $\delta(\epsilon) \to 0$ as $\epsilon \to 0$.

1.6 Organization

The rest of the paper is organized as follows: In Section 2, we present proof of Theorem 1.1. Simialrly, in Section 3 we present proof for Theorem 1.2. It is very similar to the proof of Theorem 1.1 and hence only key ingredients are presented. Finally, we present our conclusions.

2 Proof of Theorem 1.1

The proof of Theorem 1.1 essentially integrates results of [20] and [27]. The proof establishes simple connection between method of local weak convergence and convergence of max-product for MWIS (and later for MWM). Structurally, proof is divided into four steps. This four step method is divided into next four Sub-sections. Combining them provides the proof of Theorem 1.1. We note that this four step method is quite general and should be useful in providing convergence and correctness of min-sum (or max-product) algorithm for other questions.

[1] Here, the probability distribution is induced by the choice of random weights.

[2] Here, the probability distribution is induced by the choice of random edge weights.

2.1 Min-Sum and Computation Tree

We first introduce a very useful concept of computation tree. The computation tree provides a graphical interpretation of the min-sum belief at a node, say i, at a particular time, say k, in terms of initial messages and the graph structure. As we shall see, it is key to our proof. To this end, consider a fixed node i. Let T_i^k be the level-k unrolled tree corresponding to i, defined as follows: T_i^k is a weighted tree of height $k+1$, having node i as a root. All nodes have labels from the set $\{1,\ldots,n\}$ corresponding to the n nodes of the original graph. The tree is constructed according to the following recursive rule: (a) root has label i; (b) root i has a distinct child corresponding to each of its neighbors and these children get label from the original graph; and (c) a non-leaf node, say j, with parent ℓ has children corresponding to each node in $\mathcal{N}(i)\backslash\{\ell\}$ with corresponding label. The node with label j is assigned weight w_j.

The T_i^k is often called the level-k *unwrapped graph* at node i corresponding to the GM under consideration. The unwrapped graph in general is constructed by replicating the pairwise compatibility functions $\psi_{ij}(r,s)$ and potentials $\phi_i(r)$, while preserving the local connectivity of the (possibly loopy) graph. They are constructed so that the messages received by node i after k iterations in the actual graph are equivalent to those that would be received by the root i in the unwrapped graph, if the messages are passed up along the tree from the leaves to the root.

Let $t_i^k(0)$ (respectively $t_i(1)$) be the weight of maximum weight independent set in T_i^k such that the root i is not present (respectively root i is present). Now, we state the following important lemma that connects the belief of min-sum algorithm with the above defined computation tree.

Lemma 2.1. *At the end of the k^{th} iteration of the min-sum algorithm, the belief at node i of G is precisely $b_i^k = [t_i^k(0), t_i^k(1)]^T$.*

Proof. It is known [20] that under the min-sum (or max-product) algorithm, the vector b_i^k corresponds to the correct marginals for the root i of the MAP assignment on the GM corresponding to T_i^k. The pairwise compatibility functions force the MAP assignment on this tree to be an independent set.

By definition, the first (respectively second) marginal of b_i^k corresponds to the weight or likely-hood of independent set in which i is absent (respectively present). The non-normalized min-sum algorithm considered in this paper makes the exact value of the beliefs being equal to the weight of independent set. This completes the proof of Lemma 2.1.

Alternatively, the Lemma 2.1 can be easily proved using Mathematical Induction on k. □

2.2 Computation Tree and Local Topology

Consider a fixed node i in graph G, as before. Consider $V^k \subset V$ defined as

$$V_i^k = \{j \in V : \text{there is a path between } i \text{ and } j \text{ of length no more than } k\}.$$

Let $E_i^k \subset E$ be set of edges incident on these vertices. Let $G_i^k = (V_i^k, E_i^k)$ denote the subgraph of G thus created. The following Lemma relates G_i^k with the computation tree T_i^k.

Lemma 2.2. *If G_i^k is a tree for a node i, then computation tree corresponding to node i till iteration k, T_i^k is identical to G_i^k.*

Proof. By definition, the nodes and edges present in T_i^k correspond to some nodes and edges in G_i^k. The way T_i^k is constructed, all nodes that are within path-length of k are present in T_i^k. Given this, it is an easy to check (and well-known) fact that when G_i^k is a tree, the T_i^k s also a tree with identical graph structure. This completes the proof of Lemma 2.2. □

Next, we present some conditions that ensure that G_i^k is a tree.

Lemma 2.3. *Consider a fixed node i of graph G and a finite k. The following are set of different conditions that ensure that, G_i^k is a tree.*

(a) If the size of the smallest cycle of graph G is at least $2k+2$, the G_i^k is tree.

(b) If $G = G_r(n)$ with $r \geq 0$, then given k and for any $\epsilon > 0$, there exists large enough $n(\epsilon)$ such that for $n \geq n(\epsilon)$ the G_i^k is tree with probability at least $1-\epsilon$.

(c) If $G = G(n, c/n)$ with $c \geq 0$, then given k and for any $\epsilon > 0$, there exists large enough $n(\epsilon)$ such that for $n \geq n(\epsilon)$ the G_i^k is tree with probability at least $1-\epsilon$.

Proof. We first prove (a) and then provide references for (b) and (c).

Proof of (a). Suppose G_i^k is not a tree. First note that, G_i^k, by definition is a connected graph. Since, we have assumed that it is not a tree, here exists a cycle in G_i^k, say $\mathcal{C}$. Next, we show existence of cycle of length at most $2k+1$ and thus contradicting our assumption. To this end, let $u, v \in V_i^k$ be some two nodes adjacent in $\mathcal{C}$. By definition, there exists paths P_u and P_v of length at most k starting from i to nodes u and v respectively. Using edges of paths P_u, P_v and (u, v), it is straightforward to show existence of a cycle of length no more than $2k+1$. But this contradicts with the property that G (and hence G_i^k) does not have any cycle of length less than $2k+2$. Hence, our assumption of G_i^k not being tree is false.
Reference for (b) and (c). The (b) follows from result of [29] and (c) follows from result of [30].

□

2.3 Min-Sum Beliefs and Bonuses

In this section, we relate the min-sum beliefs with quantity called *bonus* – quantitative measure of advantage of including a node in Independent set on tree-graphs – originally introduced by Aldous [3] and subsequently utilized by Gamarnik et. al. [27] and others.

To this end, consider an n node finite rooted tree, H, with node 0 as its root. Let n nodes of H be numbered $0, \ldots, n-1$. Let the set of children of node i be denoted by $C(i)$. Let $H(i)$ denote the subtree rooted at i (hence, $H(0) = H$). Let $w^{H(i)}$ denote maximum weight of an independent set in $H(i)$. Define bonus of a node i (or sub-tree $H(i)$) as

$$B_{H(i)} = w^{H(i)} - \sum_{j \in C(i)} w^{H(j)}.$$

If $C(i)$ is empty, that is i is a leaf node then $B_{H(i)} = w^{H(i)} = w_i$. The above definition implies that $B_{H(i)}$ is the difference between weight of maximum weight Independent set in $H(i)$ and the weight of maximum weight independent set in $H(i)$ not containing the root i. Intuitively, $B_{H(i)}$ captures the *bonus* of including i in the candidate maximum weighted independent set of $H(i)$. Now, we state the following Lemma,

Lemma 2.4 (Lemma 7, [27]). *The bonus at node i, $B_{H(i)}$ can be recursively evaluate as*

$$B_{H(i)} = \max\left(0, w_i - \sum_{j \in C(i)} B_{H(j)}\right).$$

If $C(i)$ is empty then $B_{H(i)} = w_i$. Further, if $w_i > \sum_{j \in C(i)} B_{H(j)}$ $\left(\text{respectively} w_i < \sum_{j \in C(i)} B_{H(j)}\right)$ then all maximum weighted independent set in sub-tree $H(i)$ must contain i (respectively must not contain i).

Proof. The above Lemma follows from definition. For completeness, we refer reader to [27] for the proof.

□

Next, we state a result that relates bonus and min-sum beliefs.

Lemma 2.5. *Consider a node i with T_i^k as its computation tree and $[t_i^k(0), t_i^k(1)]^T$ as its min-sum beliefs at the end of iteration k. Then,*

$$B_{T_i^k} = \max\left(0, t_i^k(1) - t_i^k(0)\right),$$

where $B_{T_i^k}$ be the bonus for T_i^k as defined above.

Proof. Note that T_i^k is a tree, by construction. Hence, $B_{T_i^k}$ is well-defined. The key to the proof of this Lemma is: (1) definition of $B_{T_i^k}$ that it is the difference between weight of maximum independent set and weight of maximum independent set not containing i, and (2) Lemma 2.1.

Now, if $t_i^k(1) \leq t_i^k(0)$ then by Lemma 2.1, the difference between weight of maximum independent set and weight of maximum independent set not containing i is non-positive. Hence, $B_{T_i^k}$ should be 0. Similarly, if $t_i^k(1) > t_i^k(0)$ then by Lemma 2.1 and from definition of $B_{T_i^k}$, the $(t_i^k(1) - t_i^k(0))$ is the same as $B_{T_i^k}$. Putting this together, we obtain that $B_{T_i^k} = \max\left(0, t_i^k(1) - t_i^k(0)\right)$. □

The Lemma 2.5 establishes the following crucial relation between bonus and min-sum belief: convergence of min-sum beliefs (in terms of its conclusion for finding maximum independent set) is equivalent to convergence of bonuses on computation trees (of growing size). We wish to note that similar convergence on computation tree for general loopy belief propagation algorithm was studied in [28] (see Proposition 3.1, for example).

2.4 Spatial Independence and Convergence

In this section, we establish the asymptotic independence between the bonus of computation tree at the root and the initial messages in computation tree as long as the minimum length of cycle in underlying graph G is large enough (growing to ∞ asymptotically). In particular, we are interested in graph G where $G = G(n, c/n)$ for $c \leq 2e$ or $G = G_r(n)$ for $r \leq 4$ and node weight distribution is exponential of mean 1.

Consider a node, say i. Let its computation tree be T_i^k for a large k. Now, consider the top subtree of T_i^k of odd depth $d < k$ (usually $d \ll k$), denoted by $T_i^k(d)$. Thus $T_i^k(d)$ can be obtained by removing all nodes and edges of T_i^k that are beyond depth d from its root. Note that, $T_i^k(d)$ is identical to T_i^d. Hence, in what follows we use $T_i^k(d)$ and T_i^d interchangeably.

Now, consider a particular instance of node weights for all nodes in the $T_i^k \backslash T_i^k(d)$. Let they be denoted by vector $\bar{W}$. Now, consider nodes that are leaves in $T_i^k(d)$, denoted by $\delta(T_i^k(d))$. Consider one such leaf, say $\ell \in \delta(T_i^k(d))$. This leaf, ℓ, has a computation tree of depth $k - d$ underneath itself. Given $\bar{W}$, the messages coming to ℓ (under min-sum algorithm) from its children are a function of $\bar{W}$. From Lemma 2.5, we can conclude that the node ℓ can determine its bonus of node, B_ℓ, with respect to the sub-tree rooted at ℓ using the messages coming from its children and its own weight, W_ℓ. An application of Lemma 2.4 implies that $B_\ell \in [0, W_\ell]$. Let the vector of bonus values on the leaf nodes of $T_i^k(d)$ be denoted by $\mathbf{b}(d)$ and let $\mathcal{B}(d)$ represent the (compact and bounded) set of all $\mathbf{b}(d)$ such that each component of vector is bounded between 0 and the corresponding node weight. Note that since node weights are random variables, $\mathcal{B}(d)$ is a random set. From above discussion, it is clear that irrespective of graph structure and node weight for sub-graph $T_i^k \backslash T_i^k(d)$, the vector of boundary bonus values is in $\mathcal{B}(d)$. Also, Lemma 2.4 implies that the bonus at root of T_i^k, $B_{T_i^k}$ is independent of everything in $T_i^k \backslash T_i^k(d)$, given $\mathbf{b}(d)$. Thus, to prove convergence of $B_{T_i^k}$, it is sufficient to show that $B_{T_i^k}$ converges for any $\mathbf{b}(d) \in \mathcal{B}(d)$, . In what follows, our eventual goal will be to show that $Z_i^k = \mathbf{1}_{\{B_{T_i^k} > 0\}}$ converges for any $\mathbf{b}(d) \in \mathcal{B}(d)$ to prove Theorem 1.1.

Before we proceed further, we present some more usesful notation. Let the boundary condition of all bonus at leaf nodes of $T_i^k(d)$ being 0 be represented by $\mathbf{0}(d)$ and the boundary condition of all bonus being

equal to the node weight be denoted by $\mathbf{W}(d)$. For any boundary condition $\mathbf{b}(d) \in \mathcal{B}(d)$, let $\Omega_{\mathbf{b}(d)}(t)$ denote the event the bonus at root of T_i^k, $B_{T_i^k}$ is at most t given the boundary bonus as $\mathbf{b}(d) \in \mathcal{B}(d)$. From context, it should be clear that $\Omega_{\cdot}(\cdot)$ refers to a fixed d. The following result is a direct adaptation of a result of [27]. For completeness, we will give an idea for the proof.

Lemma 2.6. *Consider a fixed $t \in \mathbb{R}_+$, any odd d and for any bonus boundary condition $\mathbf{b}(d) \in \mathcal{B}(d)$,*

$$\Omega_{\mathbf{0}(d)}(t) \subseteq \Omega_{\mathbf{b}(d)}(t) \subseteq \Omega_{\mathbf{W}(d)}(t). \tag{15}$$

Further, for any $\epsilon > 0$ there exists large enough $d(\epsilon)$ such that for any odd $d \geq d(\epsilon)$ when G_i^d – the subgraph of $G = G(n, c/n)$ for $c \leq 2e$ or $G = G_r(n)$ for $r \leq 4$ – is a tree, under the probability distribution induced by randomness of node weights,

$$\Pr\left(\Omega_{\mathbf{0}(d)}(t)\right) \leq \Pr\left(\Omega_{\mathbf{W}(d)}(t)\right) \leq \Pr\left(\Omega_{\mathbf{0}(d)}(t)\right) + \epsilon. \tag{16}$$

Proof. We first prove (15). Let $\mathbf{b}(d) \in \mathcal{B}(d)$ be some boundary bonus condition for tree T_i^d. Let $\mathbf{b}(d)_1 \in \mathcal{B}(d)$ and $\mathbf{b}(d)_2 \in \mathcal{B}(d)$ be two other boundary conditions such that each component of $\mathbf{b}(d)_1$ is smaller than $\mathbf{b}(d)$; each component of $\mathbf{b}(d)$ is smaller than $\mathbf{b}(d)_2$. Let $[B_{T_i^d}|\mathbf{b}]$ denote value of bonus at the root of T_i^d given boundary bonus condition $\mathbf{b} \in \mathcal{B}(d)$.

Now, as given in the statement of Lemma, let $d \geq 0$ be odd. By definition, T_i^k is a tree and hence the bonus for each node of T_i^k is defined recursively as stated in Lemma 2.4. In particular, $[B_{T_i^d}|\mathbf{b}]$ can be recursively evaluated given the boundary bonus condition, $\mathbf{b} \in \mathcal{B}(d)$, and weights of nodes in T_i^d. Given this recursive relation, it is straightforward to verify that the value of $B_{T_i^d}$ is anti-monotone function of boundary condition for odd d, that is,

$$[B_{T_i^d}|\mathbf{b}(d)_1] \;\geq\; [B_{T_i^d}|\mathbf{b}(d)] \;\geq\; [B_{T_i^d}|\mathbf{b}(d)_2]. \tag{17}$$

To conclude the proof of (15), in addition to (17), we need the following: (1) Any boundary condition, $\mathbf{b}(d) \in \mathcal{B}(d)$ is component-wise large than $\mathbf{0}(d)$ and component-wise smaller than $\mathbf{W}(d)$; and (2) $B_{T_i^d}$ given boundary condition $\mathbf{b}(d) \in \mathcal{B}(d)$ is the same as $B_{T_i^k}$ (for $k \geq d$), given $\mathbf{b}(d)$ as the boundary condition for $T_i^k(d)$. Putting these together, we obtain that for any odd d,

$$\Omega_{\mathbf{0}(d)}(t) \subseteq \Omega_{\mathbf{b}(d)}(t) \subseteq \Omega_{\mathbf{W}(d)}(t).$$

Next we prove (16). For this suppose that given $\epsilon > 0$, there is large enough $d(\epsilon)$ (determined later) such that for some odd $d \geq d(\epsilon)$, G_i^d is a tree. Though, under the statement of Lemma, we are provided the condition of G_i^d being tree, it is useful to keep in mind that the condition of G_i^d being tree is satisfied with probability at least $1 - \epsilon$ for large enough n when graph $G = G_r(n), r \geq 2$ or $G(n, c/n), c > 0$, as stated in Lemma 2.3.

Given that G_i^d is tree, Lemma 2.2 implies that $G_i^d = T_i^d = T_i^k(d)$. Further, boundary condition $\mathbf{b}(d) \in \mathcal{B}(d)$ can be seen as a boundary condition for nodes of G_i^d. By definition of bonus, the bonus of i in G_i^d is the same as the bonus on T_i^d given the identical boundary condition $\mathbf{b}(d) \in \mathcal{B}(d)$ and the identical graph structure (and weights).

Let $\tilde{\Omega}_{\cdot}(\cdot)$ denote the event for bonus of i in G_i^d, similar to the event $\Omega_{\cdot}(\cdot)$ defined earlier for bonus of i in T_i^d. Now, results of [27] (Theorems 3 and 9) immediately imply that for large enough odd $d \geq d(\epsilon)$,

$$\Pr\left(\tilde{\Omega}_{\mathbf{0}(d)}(t)\right) \leq \Pr\left(\tilde{\Omega}_{\mathbf{W}(d)}(t)\right) \leq \Pr\left(\tilde{\Omega}_{\mathbf{0}(d)}(t)\right) + \epsilon. \tag{18}$$

Repeating what we stated above, that is, $G_i^d = T_i^d$, $\tilde{\Omega}_{\cdot}(\cdot) = \Omega_{\cdot}(\cdot)$. Hence, we obtain that for large enough odd $d \geq d(\epsilon)$

$$\Pr\left(\Omega_{\mathbf{0}(d)}(t)\right) \leq \Pr\left(\Omega_{\mathbf{W}(d)}(t)\right) \leq \Pr\left(\Omega_{\mathbf{0}(d)}(t)\right) + \epsilon. \tag{19}$$

This completes the proof of Lemma 2.6. □

2.5 Putting Things Together

In this section, we complete the proof of Theorem 1.1. Define

$$\hat{\Omega}_- = \Omega_{\mathbf{0}(d)}(0), \text{ and } \hat{\Omega}_+ = \Omega^c_{\mathbf{W}(d)}(0).$$

We state the following two Lemmas.

Lemma 2.7. *Given odd d, for any boundary bonus condition $\mathbf{b}(d) \in \mathcal{B}(d)$, the $B_{T_i^k} > 0$ for any $\omega \in \hat{\Omega}_+$ and $B_{T_i^k} = 0$ for any $\omega \in \hat{\Omega}_-$.*

Proof. It follows directly from definitions of $\hat{\Omega}_+$, $\hat{\Omega}_-$ and Lemma 2.6. □

Lemma 2.8. *Consider a fixed node i. Under the setup of Lemma 2.6 (equivalently that of Theorem 1.1), for any $\epsilon > 0$, there exists large enough $n(\epsilon)$ and $k(\epsilon)$ such that for (random) graph with $n \geq n(\epsilon)$, the x_i^k of min-sum algorithm converges to the correct value for $k > k(\epsilon)$ with probability[3] at least $1 - \epsilon$.*

Proof. Consider $Z_i^k = \mathbf{1}_{\{B_{T_i^k} > 0\}}$. Under the setup of Lemma 2.8, we first show equivalence between convergence of Z_i^k and x_i^k. Since weights are distributed according to exponential random variable, the probability of W_i being equal to the sum of bonuses of nodes that are children of i in T_i^k is 0. Hence, from Lemma 2.4, with probability 1 either i belongs to all maximum weight Independent set in T_i^k or it does not belong all maximum weight Independent set in T_i^k. Consequently, another use of Lemma 2.4 implies that Z_i^k is an indicator of event that i belongs to all maximum weight Independent set. Now i belongs to or does not belong to all maximum weight independent set with probability 1. Hence Lemma 2.1 implies that $t_i^k(0) \neq t_i^k(1)$ with probability 1. Hence, Lemma 2.1 and definition of x_i^k implies that $x_i^k = Z_i^k$ with probability 1. Hence, in order to prove convergence, it is sufficient to prove that there exists $k(\epsilon)$ such that for $k \geq k(\epsilon)$, Z_i^k converges with probability at least $1 - \epsilon$. Next, we use Lemmas 2.6 and 2.7 to do so.

From Lemma 2.7, for $k > d$ the Z_i^k converges on set $\hat{\Omega} = \hat{\Omega}_+ \cup \hat{\Omega}_-$ as defined above. Now, it is sufficient to show that probability of $\hat{\Omega}$ is at least $1 - \epsilon$. To this end, consider the following. Consider $d(\epsilon/2)$ as in Lemma 2.6. Consider smallest odd $d \geq d(\epsilon/2)$. From Lemma 2.3, there exists large enough $n(\epsilon)$ such that for $n \geq n(\epsilon)$, for a given node i the G_i^d (either $G_r(n)$ or $G(n, c/n)$) is tree with probability at least $1 - \epsilon/2$. That is,

$$\Pr\left(G_i^d \text{ is not tree}\right) \quad \leq \quad \epsilon/2. \tag{20}$$

For any odd d, $\Omega_{\mathbf{0}(d)}(0) \subseteq \Omega_{\mathbf{W}(d)}(0)$ and hence by Lemma 2.6, given that $d \geq d(\epsilon/2)$,

$$\Pr\left(\Omega_{\mathbf{W}(d)}(0) \backslash \Omega_{\mathbf{0}(d)}(0) | G_i^d \text{ is tree}\right) \quad \leq \quad \epsilon/2. \tag{21}$$

From (20) and (21) it immediately follows that

$$\Pr\left(\Omega_{\mathbf{W}(d)}(0) \backslash \Omega_{\mathbf{0}(d)}(0)\right) \quad \leq \quad \epsilon. \tag{22}$$

Now, consider the following.

$$\begin{aligned}
\Pr(\hat{\Omega}) &= \Pr(\hat{\Omega}_+ \cup \hat{\Omega}_-) = \Pr(\hat{\Omega}_+) + \Pr(\hat{\Omega}_- \cap \hat{\Omega}_+^c) \\
&= \Pr(\hat{\Omega}_+) + \Pr(\hat{\Omega}_-) \qquad (23) \\
&= \Pr\left(\Omega^c_{\mathbf{W}(d)}(0)\right) + \Pr\left(\Omega_{\mathbf{0}(d)}(0)\right) \\
&= 1 - \left\{\Pr\left(\Omega_{\mathbf{W}(d)}(0)\right) - \Pr\left(\Omega_{\mathbf{0}(d)}(0)\right)\right\}. \\
&= 1 - \Pr\left(\Omega_{\mathbf{W}(d)}(0) \backslash \Omega_{\mathbf{0}(d)}(0)\right). \\
&\geq 1 - \epsilon, \qquad (24)
\end{aligned}$$

[3] Recall that the probability is induced by random graph and random node weights.

where (24) follows from (22) and the (23) follows from $\Omega_{\mathbf{0}(d)}(0) \subseteq \Omega_{\mathbf{W}(d)}(0)$ due to Lemma 2.6. This shows that x_i^k converges with probability at least $1-\epsilon$ for large enough k and graph size n. To prove the correctness of x_i^k note the following: to obtain (22) and subsequently (24), we have used the fact that the neighborhood of node i till depth d, G_i^d is tree and hence identical to $T_i^k(d)$ or T_i^d. Now, under event $\hat{\Omega}$, the bonus value at node i in T_i^k or G_i^d is determined by the graph structure (including weights) T_i^d or G_i^d. Since $G_i^d = T_i^d$ and event $\hat{\Omega}$ holds, the positivity of bonus values in G_i^d is the same as in T_i^d (for all boundary conditions). This establishes the correctness of x_i^k as it is indicator of bonus being positive or zero. This completes the proof of Lemma 2.8.

We note that the value of $k(\epsilon)$ can be set at the smallest odd $d \geq d(\epsilon/2)$, while $n(\epsilon)$ is determined by the need of (20). □

Finally, we wrap up the proof of Theorem 1.1 as follows.

Proof of Theorem 1.1. We first present the proof of (a) and then proof of (b).

Proof of (a). Consider any node, say i of graph G. The Lemma 2.8 shows that for any $\epsilon > 0$, there exists large enough $n(\epsilon), k(\epsilon)$ such that for graph with nodes $n \geq n(\epsilon)$, the decision variable of min-sum at node i, x_i^k, converges for $k \geq k(\epsilon)$ with probability at least $1-\epsilon$. This completes the proof of convergence as claimed in Theorem 1.1(a).

Proof of (b). Consider the decisions of min-sum algorithm in terms of $\mathcal{I}^k$ at the end of iteration k. Using Lemma 2.8 and Markov's inequality, we obtain that for $k \geq k(\epsilon^2/4)$, the size of the symmetric difference of $\mathcal{I}^k$ and $\mathcal{I}^*$, $\mathcal{I}^k \Delta \mathcal{I}^*$, is at most ϵn with probability at least $1-\epsilon/4$. Define a node as a "bad" node if it belongs to $\mathcal{I}^k$ but does not belong to $\mathcal{I}^*$. A node that is not "bad" is "good". By definition, number of bad nodes is no more than size of the set $\mathcal{I}^k \Delta \mathcal{I}^*$. Hence, number of bad nodes is no more than ϵn under the above setup.

By definition two "good" nodes can not be neighbors as well as present in $\mathcal{I}^k$. Hence, under the modification procedure described before the statement of Theorem 1.1, removal of each node from $\mathcal{I}^k$ to obtain eventual $\hat{\mathcal{I}}^k$, can be associated with the presence of a "bad" neighbor. Using this property, the difference between $\hat{\mathcal{I}}^k$ and $\mathcal{I}^*$ can be bounded by the size of neighborhood[4] of any ϵn nodes in G. For $G_r(n)$, it is no more than $r\epsilon n$. For $G(n, c/n)$, using the property that for large n, the number of neighbors of each node is like Poisson(c), we can show that the neighborhood of any ϵn nodes is no more than $\epsilon_1 n$ with probability at least $1-\epsilon/4$, where $\epsilon_1 \to 0$ as $\epsilon \to 0$. Thus, till now we have obtained that for some $\epsilon_2 > 0$ such that $\epsilon_2 \to 0$ as $\epsilon \to 0$,

$$\Pr\left(|\hat{\mathcal{I}}^k \Delta \mathcal{I}^*| \geq \epsilon_2 n\right) \quad \leq \quad \epsilon/2. \tag{25}$$

Now, using property of exponential variables, the weight of any $\epsilon_2 n$ nodes can be upper bounded by $\epsilon_3 n$ with probability at least $1-\epsilon/4$, where again $\epsilon_3 \to 0$ as $\epsilon_2 \to 0$. Also, it is easy to see that both in $G_r(n)$ with $r \leq 4$ or $G(n, c/n)$ with $c \leq 2e$, the weight of $\mathcal{I}^*$ is at least αn, for some constant $\alpha > 0$, with probability at least $1-\epsilon/4$ for large enough n. From (25) and above discussion, it is easy to see that for some $\delta(\epsilon) > 0$ where $\delta(\epsilon) \to 0$ as $\epsilon \to 0$,

$$P\left(\frac{|W(\hat{\mathcal{I}}^k) - W(\mathcal{I}^*)|}{W(\mathcal{I}^*)} \geq \delta(\epsilon)\right) \quad \leq \quad \epsilon. \tag{26}$$

This completes the proof of Theorem 1.1. □

[4]Here, the neighborhood of a node $v \in G$ is the set of its immediate neighbors in G, i.e. $\{u \in G : (u, v) \in E\}$.

3 Proof of Theorem 1.2

The proof of Theorem 1.2 is very similar to that of Theorem 1.1. Similar to Theorem 1.1, we will use results of [20] and [27] for establishing the proof of Theorem 1.2. In what follows, the proof is described in different sub-sections with details omitted when they are similar to that presented in proof of Theorem 1.1.

3.1 Min-Sum and Computation Tree

The computation tree for min-sum algorithm for MWM is identical to that described in Section 2.1 with the following difference: the edges of computation tree are assigned weight w_{uv} if the end points of edge correspond to nodes u and v of G. For MWM, the node weights are irrelevant.

As before, let T_i^k denote the computation tree of min-sum algorithm for node i till iteration k. Let $t_i^k(j), j \in \mathcal{N}(i) \cup \{n+1\}$ denote the weight of maximum weight matching on T_i^k among all matchings under which root node i is connected to j under the matching (recall that i connected to $n+1$ means that it is not connected to any node). Let $b_i^k = [b_i^k(1), \ldots, b_i^k(n+1)]^T$ denote the belief vector at node i at the end of iteration k under min-sum algorithm for MWM. Then, we state the following result similar to Lemma 2.1.

Lemma 3.1. *For any $j \in \mathcal{N}(i) \cup \{n+1\}$,*

$$b_i^k(j) = t_i^k(j).$$

Like Lemma 2.1, the proof follows from [20].

3.2 Computation Tree and Local Topology

As in Section 2.2, with respect to a node i and $k \geq 0$, the subgraph of G, G_i^k can be defined. The Lemmas 2.2 and 2.3 hold verbatim.

3.3 Min-Sum Beliefs and Bonuses

Similar to independent set, bonus for each node, can be defined in the case of matching as well. In particular, *bonus* of a node i is the difference between weight of maximum weight matching in G and weight of maximum weight matching that does not match i to any node in G. Given this definition, lets consider bonus on trees.

To this end, consider an n node finite rooted tree, H, with node 0 as its root and its nodes numbered $0, \ldots, n-1$. Let the set of children of node i be denoted by $C(i)$. Let $H(i)$ denote the subtree rooted at i (hence, $H(0) = H$). Let edges of H be assigned non-negative weights. Let $w^{H(i)}$ denote maximum weight of matching in $H(i)$. Define bonus of i, denoted by $B_{H(i)}$, to be the difference between $w^{H(i)}$ and the maximum weight of any matching in $H(i)$ that does not allow i to be matched with any node. The following was stated and proved in [27] (earlier, considered by [3]).

Lemma 3.2 (Lemma 8, [27]). *The bonus at node i, $B_{H(i)}$ can be recursively evaluate as*

$$B_{H(i)} = \max\left(0, \max_{j \in C(i)} (w_{i,j} - B_{H(j)})\right).$$

If $C(i)$ is empty then $B_{H(i)} = 0$. Further, if $w_{i,j} - B_{H(j)} > w_{i,j'} - B_{H(j')}$, $\forall j' \in C(i) \backslash \{j\}$ and $w_{i,j} - B_{H(j)} > 0$ then every maximum weight matching in $H(i)$ contains edge (i,j). If $w_{i,j} - B_{H(j)} < 0$, $\forall j \in C(i)$, then all maximum weight matchings in $H(i)$ do not contain any edge incident on i.

Next, we state a result that relates bonus and min-sum beliefs.

Lemma 3.3. *Consider a node i with T_i^k as its computation tree and $[t_i^k(1), \ldots, t_i^k(n+1)]$ be its min-sum beliefs at the end of iteration k. Then,*

$$B_{T_i^k} = \max\left(0, \max_{j \in \mathcal{N}(i)} (t_i^k(j) - t_i^k(n+1))\right),$$

where $B_{T_i^k}$ be the bonus for T_i^k as defined above.

Proof. Note that T_i^k is a tree, by construction. Hence, $B_{T_i^k}$ is well-defined as above. The key to the proof of this Lemma is: (1) definition of $B_{T_i^k}$ that it is the difference between weight of maximum weight matching and weight of maximum independent set not containing i, and (2) Lemma 3.1.

Now, if for all $j \in \mathcal{N}(i)$, $t_i^k(j) \leq t^k(n+1)$ then by Lemma 3.1 and definition of bonus, $B_{T_i^k} = 0$. Else, under the maximum weight matching in T_i^k, root node i must be connected to j^*, where $j^* = \arg\max_{j \in \mathcal{N}(i)} t_i^k(j)$. Hence, the bonus $B_{T_i^k}$ should be equal to $t_i^k(j^*) - t_i^k(n+1)$. This completes the proof of Lemma 3.3. □

3.4 Spatial Independence and Convergence

In this section, we use results of [27] to establish asymptotic independence between the bonus of computation tree at the root and the initial messages in computation tree. In particular, we are interested in graph G where $G = G(n, c/n)$ for $c > 0$ or $G = G_r(n)$ for $r \geq 2$ and node weight distribution is exponential of mean 1.

As in Lemma 2.2 and proof of Theorem 1.1, we will consider G ($G(n, c/n)$ or $G_r(n)$) with large enough n so that the d depth neighborhood of node i, G_i^d is a tree. This happens with probability at least $1 - \epsilon/2$ for appropriate choice of n given d. Henceforth, we assume that for our choice of d and ϵ, n is chosen to be large enough.

Given this, consider computation tree T_i^k for a large k. Now, consider the top subtree of T_i^k of odd depth $d < k$, denoted by $T_i^k(d)$, which is the same as T_i^d. The Lemma 3.2 suggests that if bonus values at the boundary nodes of $T_i^k(d)$ is fixed and edge weights for $T_i^k(d)$ are known then $B_{T_i^k}$ can be determined without knowledge of everything in $T_i^k \backslash T_i^k(d)$. Further, by Lemma 3.2 bonus at a node j is non-negative and no larger than the maximum of the edge weights incident on it. Let W_j^* denote this quantity for node j. Let $\mathcal{B}(d)$ denote the set of vectors representing boundary bonus condition for $T_i^k(d)$ with the component of boundary condition, corresponding to a boundary node j, is between $[0, W_j^*]$. As before, let $\mathbf{0}(d)$ denote boundary condition when all nodes have bonus 0 and let $\mathbf{W}^*(d)$ denote boundary condition when all nodes have maximal bonus. Let root i in T_i^k have $N_i = \mathcal{N}(i)$ children. Let the subtrees with each of these children at root be numbered $1, \ldots, N_i$. Let the boundary bonus condition for subtree j be denoted by $\mathbf{b}(d)_j$. Recall that for $G_r(n)$ $N_i = r$ while for $G(n, c/n)$ it is distributed like Poisson(c). Thus, any boundary condition in $\mathcal{B}(d)$ can also be represented as tuple $(\mathbf{b}(d)_j)_{1 \leq j \leq N_i}$. For simplicity, we present it as $(\mathbf{b}(d)_j)$. Let $\mathbf{b}(d)_j^+$, $1 \leq j \leq N_i$ denote the boundary condition when each leaf node for j^{th} subtree is conditioned to have maximal bonus while all leaf nodes in other subtree are conditioned to be 0. Similarly, let $\mathbf{b}(d)_j^-$ denote boundary condition when all boundary nodes are set to have maximal bonus value, but leaf nodes of subtree j. Now, define the following events.

(a) For any $t \geq 0$ and $\mathbf{b}(d) \in \mathcal{B}(d)$, let $\Omega_{\mathbf{b}(d)}(t)$ denote the event that bonus $B_{T_i^k}$ is at most t given boundary condition $\mathbf{b}(d)$.

(b) Let $\Omega_{j,\mathbf{b}(d)} \subset \Omega^c_{\mathbf{b}(d)}(0)$ denote the event that the bonus $B_{T_i^k}$ is positive and edge (i,j) presents maximum benefit (i.e. (i,j) should be part of MWM) given boundary condition as $\mathbf{b}(d)$.

Next, we state a result similar to Lemma 2.6. We omit proof as it is similar to that of Lemma 2.6 and uses result (Theorems 3 and 9) of [27] for matching.

Lemma 3.4. *Consider a fixed $t \in \mathbb{R}_+$, any odd d and for any bonus boundary condition $\mathbf{b}(d) \in \mathcal{B}(d)$,*

$$\Omega_{\mathbf{0}(d)}(t) \subseteq \Omega_{\mathbf{b}(d)}(t) \subseteq \Omega_{\mathbf{W}^*(d)}(t), \tag{27}$$

and

$$\Omega_{j,\mathbf{b}(d)^+_j} \subseteq \Omega_{j,\mathbf{b}(d)} \subseteq \Omega_{j,\mathbf{b}(d)^-_j}. \tag{28}$$

Further, for any $\epsilon > 0$ there exists large enough $d(\epsilon)$ such that for any odd $d \geq d(\epsilon)$ when G_i^d – the subgraph of $G = G(n, c/n)$ for $c > 0$ or $G = G_r(n)$ for $r \geq 2$ – is a tree, under the probability distribution induced by randomness of edge weights,

$$\Pr\left(\Omega_{\mathbf{0}(d)}(t)\right) \leq \Pr\left(\Omega_{\mathbf{W}(d)}(t)\right) \leq \Pr\left(\Omega_{\mathbf{0}(d)}(t)\right) + \epsilon, \tag{29}$$

and

$$\Pr\left(\Omega_{j,\mathbf{b}(d)^+_j}\right) \leq \Pr\left(\Omega_{j,\mathbf{b}(d)^-_j}\right) \leq \Pr\left(\Omega_{j,\mathbf{b}(d)^+_j}\right) + \epsilon, \tag{30}$$

3.5 Putting Things Together

Similar to proof of Theorem 1.1 presented in Section 2.5, the proof of Theorem 1.2 follows from Lemmas 3.1- 3.4. We omit details as they are identical to the proof of Theorem 1.1.

4 Conclusion

In this paper, we established convergence and correctness of the recently well studied Max Product algorithm for Maximum Weight Independent Set and Maximum Weight Matching for sparse random graphs. Our results crucially utilized results of [27].

As a conclusion of our work, we find that the following three properties are sufficient to establish the correctness (approximation with any ϵ) and convergence of max-product algorithm: (1) Locally tree-like graph, (2) Monotone property of "bonus" style function, and (3) Spatial independence or *local optimality property*.

For many questions where the method of local weak convergence seem to work, the above properties seem to hold. This suggests that for such questions, Max Product can be used as algorithm to find good solutions.

References

[1] S. M. Aji and R. J. McEliece, "The Generalized Distributive Law," *IEEE Trans. Inform. Theory*, Vol. 46, pp. 325-343, 2000.

[2] S. M. Aji, G. B. Horn and R. J. McEliece, "On the Convergence of Iterative Decoding on Graphs with a Single Cycle," *Proc. 1998 IEEE Int. Symp. Information Theory*, Cambridge, MA, p. 276, 1998.

[3] D. Aldous, "The zeta (2) Limit in the Random Assignment Problem," *Random Structures and Algorithms*, Vol. 18, pp. 381-418, 2001.

[4] D. Aldous, “Asymptotics in the Random Assignment,” Problem. Probab. Th. Related Fields, Volume 93, 1992.

[5] D. Aldous and A. Bandyopadhyay, “A Survey of Max-type Recursive Distributional Equations,” Annals of Applied Probability, Volume 15, 2005.

[6] D. Bertsekas and J. Tsitsiklis, “Parallel and Distributed Computation: Numerical Methods,” *Prentice Hall*, Englewood Cliffs, N. J., 1989.

[7] P. Brémaud, “Markov Chains, Gibbs Fields, Monte Carlo Simulation, and Queues,” *Springer*, 1991.

[8] J. Edmonds and R. Karp, “Theoretical Improvements in Algorithmic Efficiency for Network Flow Problems,” *Jour. of the ACM*, Vol. 19, pp 248-264, 1972.

[9] B. J. Frey, R. Koetter and A. Vardy, “Skewness and pseudocodewords in iterative decoding” *Proc. 1998 IEEE Int. Symp. Information Theory*, Cambridge, MA, p. 148, 1998.

[10] B.J. Frey, R. Koetter, “Exact inference using the attenuated max-product algorithm”, in *Advanced Mean Field Methods: Theory and Practice*, ed. Manfred Opper and David Saad, MIT Press, 2000.

[11] R. G. Gallager, “Low Density Parity Check Codes,” *MIT Press*, Cambridge, MA, 1963.

[12] G. B. Horn, “Iterative Decoding and Pseudocodewords,” *Ph.D. dissertation*, Dept. elect. Eng., Calif. Inst. Technol., Pasadena, CA, 1999.

[13] S. Lauritzen, “Graphical models,” *Oxford University Press*, 1996.

[14] E. Lawler, “Combinatorial Optimization: Networks and Matroids”, *Holt, Rinehart and Winston*, New York, 1976.

[15] N. McKeown, V. Anantharam and J. Walrand, “Achieving 100 % Throughput in an Input-Queued Switch,” *Infocom*, Vol. 1, pp 296-302, 1996.

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

[17] T. Richardson and R. Urbanke, “The Capacity of Low-Density Parity Check Codes under Message-Passing Decoding,” *IEEE Trans. Info. Theory*, Vol. 47, pp 599-618, 2001.

[18] M. Wainwright, T. Jaakkola and A. Willsky, “Tree Consistency and Bounds on the Performance of the Max-Product Algorithm and its Generalizations,” *Statistics and Computing*, Vol. 14, pp 143-166, 2004.

[19] M. Wainwright, M. Jordan, “Graphical models, exponential families, and variational inference,” *Tech. Report*, Dept. of Stat.,University of Cal., Berkeley, 2003.

[20] Y. Weiss, “Belief propagation and revision in networks with loops,” *MIT AI Lab.*, Tech. Rep. 1616, 1997.

[21] Y. Weiss, “Correctness of local probability propagation in graphical models with loops,” *Neural Comput.*, Vol. 12, pp. 1-42, 2000.

[22] Y. Weiss and W. Freeman, “Correctness of belief propagation in Gaussian graphical models of arbitrary topology,” *Neural Comput.*, Vol. 13, Issue 10, pp 2173-2200, 2001

[23] Y. Weiss W. Freeman, “On the optimality of solutions of the max-product belief propagation algorithm in arbitrary graphs.,” *IEEE Trans. Info. Theory*, Vol. 47, pp 736-744, 2001.

[24] J. Yedidia, W. Freeman and Y. Weiss, “Generalized Belief Propagation,” *Mitsubishi Elect. Res. Lab.*, TR-2000-26, 2000.

[25] J. Yedidia, W. Freeman and Y. Weiss, “Understanding Belief Propagation and its Generalizations,” *Mitsubishi Elect. Res. Lab.*, TR-2001-22, 2000.

[26] D Aldous and J. M. Steele,“The objective method: Probabilistic combinatorial optimization and local weak convergence,” Discrete Combinatorial Probability, H. Kesten Ed., Springer-Verlag, 2003.

[27] D. Gamarnik, T. Nowicki and G. Swirscsz, “Maximum Weight Independent Set and Matching in Sparse Random Graphs. Exact Results using the Local Weak Convergence Method,” available at arXiv.org as *arXiv:math.PR/0309441*.

[28] S. C. Tatikonda and M. I. Jordan, “Loopy Belief Propagation and Gibbs Measure,” Berkeley Working Paper, 2002.

[29] S. Janson, T. Luczak and A. Rucinski, “Random Graphs,” John Wiley and Sons Inc., 2000.

[30] J. H. Spencer, “Ten Lectures on Probabilistic Method,” Second Ed., SIAM 1994.

[31] M. Bayati, D. Shah and M. Sharma, “Maximum Weight Matching via Max Product Belief Propagation,” To appear in the proceedings of *ISIT* , 2005.

[32] R. Karp and M. Sipser, “Maximum matchings in sparse random graphs,” In proceedings of *FOCS*, 1981.

[33] D. Gamarnik,“Linear phase transition in random linear constraint satisfaction problems,” Probability Theory and Related Fields, 2004.

[34] J. Michael Steele, “Probability Theory and Combinatorial Optimization,” SIAM book, 1997.