Smoothed Analysis of Belief Propagation for Minimum-Cost Flow and Matching*

Tobias Brunsch[1], Kamiel Cornelissen[2], Bodo Manthey[2], and Heiko Röglin[1]

[1] University of Bonn, brunsch@cs.uni-bonn.de, heiko@roeglin.org
[2] University of Twente, k.cornelissen@utwente.nl, b.manthey@utwente.nl

Abstract. Belief propagation (BP) is a message-passing heuristic for statistical inference in graphical models such as Bayesian networks and Markov random fields. BP is used to compute marginal distributions or maximum likelihood assignments and has applications in many areas, including machine learning, image processing, and computer vision. However, the theoretical understanding of the performance of BP is unsatisfactory. Recently, BP has been applied to combinatorial optimization problems. It has been proved that BP can be used to compute maximum-weight matchings and minimum-cost flows for instances with a unique optimum. The number of iterations needed for this is pseudo-polynomial and hence BP is not efficient in general.
We study belief propagation in the framework of smoothed analysis and prove that with high probability the number of iterations needed to compute maximum-weight matchings and minimum-cost flows is bounded by a polynomial if the weights/costs of the edges are randomly perturbed. To prove our upper bounds, we use an isolation lemma by Beier and Vöcking (SIAM J. Comput., 2006) for matching and generalize an isolation lemma for min-cost flow by Gamarnik, Shah, and Wei (Oper. Res., 2012). We also prove almost matching lower tail bounds for the number of iterations that BP needs to converge.

1 Belief Propagation

Belief propagation (BP) is a message-passing algorithm that is used for solving probabilistic inference problems on graphical models. It has been introduced by Pearl in 1988 [8]. Typical graphical models to which BP is applied are Bayesian networks and Markov random fields. There are two variants of the BP algorithm. The sum-product variant is used to compute marginal probabilities. The max-product or min-sum variant is used to compute maximum a posteriori (MAP) probability estimates.

Recently, BP has experienced great popularity. It has been applied in a large number of fields, such as machine learning, image processing, computer vision, and statistics. For an introduction to BP and several applications, we refer to

* This research was supported by ERC Starting Grant 306465 (BeyondWorstCase) and NWO grant 613.001.023 (Smoothed Analysis of Belief Propagation). A full version of this paper is available at http://arxiv.org/abs/1211.3299.

Yedidia et al. [14]. There are basically two main reasons for the popularity of BP. First of all, it is generally applicable and easy to implement because of its simple and iterative message-passing nature. In addition, it performs well in practice in numerous applications.

If the graphical model is tree-structured, BP computes exact marginals/MAP estimates. In case the graphical model contains cycles, convergence and correctness of BP have been shown only for specific classes of graphical models. To improve the general understanding of BP and to gain new insights about it, the performance of BP as either a heuristic or an exact algorithm for several combinatorial optimization problems has been studied. Amongst others it has been applied to the maximum-weight matching (MWM) problem, the minimum spanning tree problem [3], the minimum-cost flow (MCF) problem, and the maximum-weight independent set problem [11].

Bayati et al. [4] have shown that max-product BP correctly computes the MWM in bipartite graphs in pseudo-polynomial time if it is unique. Gamarnik et al. [6] have shown that max-product BP computes the MCF in pseudo-polynomial time if it is unique.

1.1 Belief Propagation for Matching and Flow Problems

Bayati et al. [4] have shown that the max-product BP algorithm correctly computes the maximum-weight matching in bipartite graphs if it is unique. Convergence of BP takes pseudo-polynomial time and depends linearly on the weight of the heaviest edge and on $1/\delta$, where δ is the difference in weight between the best and second-best matching. In Section 2 we describe BP for MWM in detail.

Belief propagation has also been applied to finding maximum-weight perfect matchings in arbitrary graphs and to finding maximum-weight perfect b-matchings [2, 10], where a perfect b-matching is a set of edges such that every vertex is incident to exactly b edges in the set. For arbitrary graphs the BP algorithm for MWM does not necessarily converge [10]. However, Bayati et al. [2] and Sanghavi et al. [10] have shown that the BP algorithm converges to the optimal matching if the relaxation of the corresponding linear program has an optimal solution that is unique and integral. The number of iterations needed until convergence depends again linearly on the reciprocal of the parameter δ. Bayati et al. [2] have also shown that the same result holds for the problem of finding maximum-weight b-matchings that do not need to be perfect.

It turns out that BP can, to some extent, solve the relaxation of the corresponding linear program for matching, even if it has a non-integral optimal solution. Bayati et al. [2] have shown that it is possible to solve the LP relaxation by considering so-called *graph covers*, in which they compute a bipartite matching. In case of an optimum that is unique and integral, the optimal solution in the graph cover corresponds to the optimal solution. In case of a unique but fractional optimal solution, the average of the estimates of two consecutive iterations (both of which are perfect matchings in the graph cover) yield a value of 0, 1/2, or 1 for any edge, which then equals its value in the optimal solution of the relaxed LP. Sanghavi et al. [10] have shown that BP remains uninformative

for some edges (and outputs "?" for those), but computes the correct values for all edges that have a fixed integral value in all optimal solutions.

Gamarnik et al. [6] have shown that BP can be used to find a minimum-cost flow, provided that the instance has a unique optimal solution. The number of iterations until convergence is pseudo-polynomial and depends again linearly on the reciprocal of the difference in cost between the best and second-best integer flow. In addition, they have proved a discrete isolation lemma [6, Theorem 8.1] that shows that the edge costs can be slightly randomly perturbed to ensure that, with probability at least 1/2, the perturbed MCF instance has a unique optimal solution. Using this result, they have constructed an FPRAS for MCF using BP.

1.2 Smoothed Analysis

Smoothed analysis has been introduced by Spielman and Teng [12] in order to explain the performance of the simplex method for linear programming. It is a hybrid of worst-case and average-case analysis and an alternative to both: An adversary specifies an instance, and this instance is then slightly randomly perturbed. The perturbation can, for instance, model noise from measurement. Since its invention in 2001, smoothed analysis has been applied in a variety of contexts. We refer to two recent surveys [7, 13] for a broader picture.

We apply smoothed analysis to BP for min-cost flow and maximum-weight matching. To do this, we consider the following general probabilistic model.

- The adversary specifies the graph $G = (V, E)$ and, in case of min-cost flow, the integer capacities of the edges and the integer budgets (both are not required to be polynomially bounded). Additionally the adversary specifies a probability density function $f_e : [0, 1] \to [0, \phi]$ for every edge e.
- The costs (for min-cost flow) or weights (for matching) of the edges are then drawn independently according to their respective density function.

The parameter ϕ controls the adversary's power: If $\phi = 1$, then we have the average case. The larger ϕ, the more powerful the adversary. The role of ϕ is the same as the role of $1/\sigma$ in the classic model of smoothed analysis, where instances are perturbed by independent Gaussian noise with standard deviation σ. In that model the maximum density ϕ is proportional to $1/\sigma$.

1.3 Our Results

We prove upper and lower tail bounds for the number of iterations that BP needs to solve maximum-weight matching problems and min-cost flow problems. Our bounds match up to a small polynomial factor.

In Sections 3 and 4 we prove that the probability that BP needs more than t iterations is bounded by $O(n^2 m\phi/t)$ for the min-cost flow problem and $O(nm\phi/t)$ for various matching problems, where n and m are the number of nodes and edges of the input graph, respectively. The upper bound for matching problems holds for the variants of BP for the maximum-weight matching

problem in bipartite graphs [4] as well as for the maximum-weight (perfect) b-matching problem in general graphs [2, 10]. For the latter it is required that the polytope corresponding to the relaxation of the matching LP is integral. If this is not the case, we can still solve the relaxation of the matching LP with a slightly modified BP algorithm [2] using graph covers (see the comments at the end of Section 4.1). To prove the upper tail bound for BP for MCF we use a continuous isolation lemma that is similar to the discrete isolation lemma by Gamarnik et al. [6, Theorem 8.1]. We need the continuous version since we do not only want to have a unique optimal solution, but we also need to quantify the gap between the best and the second-best solution.

These upper tail bounds are not strong enough to yield any bound on the expected number of iterations. Indeed, in Section 5 we show that this expectation is not finite by providing a lower tail bound of $\Omega(n\phi/t)$ for the probability that t iterations do not suffice to find a maximum-weight matching in bipartite graphs. This lower bound even holds in the average case, i.e., if $\phi = 1$, and it carries over to the variants of BP for the min-cost flow problem and the minimum/maximum-weight (perfect) b-matching problem in general graphs mentioned above [2, 4, 6, 10]. The lower bound matches the upper bound up to a factor of $O(m)$ for matching and up to a factor of $O(nm)$ for min-cost flow. The smoothed lower bound even holds for complete (i.e., non-adversarial) bipartite graphs.

Finally, let us remark that, for the min-cost flow problem, we bound only the number of iterations that BP needs until convergence. The messages might be super-polynomially long. For all matching problems, however, the size of each message is polynomial in the input size and linear in the number of iterations.

2 Definitions and Problem Statement

2.1 Maximum-Weight Matching and Minimum-Cost Flow

First we define the maximum-weight matching problem on bipartite graphs. Consider an undirected weighted bipartite graph $G = (U \cup V, E)$ with $U = \{u_1, \ldots, u_n\}$, $V = \{v_1, \ldots, v_n\}$, and $E \subseteq \{(u_i, v_j) = e_{ij}, 1 \leq i, j \leq n\}$. Each edge e_{ij} has weight $w_{ij} \in \mathbb{R}^+$. A collection of edges $M \subseteq E$ is called a matching if each node of G is incident to at most one edge in M. We define the weight of a matching M by $w(M) = \sum_{e_{ij} \in M} w_{ij}$. The maximum-weight matching $M^\star$ of G is defined as $M^\star = \operatorname{argmax}\{w(M) \mid M \text{ is a matching of } G\}$.

A b-matching $M \subseteq E$ in an arbitrary graph $G = (V, E)$ is a set of edges such that every node from V is incident to at most b edges from M. A b-matching is called perfect if every node from V is incident to exactly b edges from M. Also for these problems we assume that each edge $e \in E$ has a certain weight w_e and we define the weight of a b-matching M accordingly.

In the min-cost flow problem (MCF), the goal is to find a cheapest flow that satisfies all capacity and budget constraints. We are given a graph $G = (V, E)$ with $V = \{v_1, \ldots, v_n\}$. In principle we allow multiple edges between a pair of nodes, but for ease of notation we consider simple directed graphs. Each node v

has a budget $b_v \in \mathbb{Z}$. Each directed edge $e = e_{ij}$ from v_i to v_j has capacity $u_e \in \mathbb{N}_0$ and cost $c_e \in \mathbb{R}^+$. For each node $v \in V$, we define E_v as the set of edges incident to v. For each edge $e \in E_v$ we define $\Delta(v, e) = 1$ if e is an out-going edge of v and $\Delta(v, e) = -1$ if e is an in-going edge of v. In the MCF one needs to assign a flow f_e to each edge e such that the total cost $\sum_{e \in E} c_e f_e$ is minimized and the flow constraints $0 \leq f_e \leq u_e$ for all $e \in E$, and budget constraints $\sum_{e \in E_v} \Delta(v, e) f_e = b_v$ for all $v \in V$ are satisfied. We refer to Ahuja et al. [1] for more details.

Let us remark that we could have allowed also rational values for the budgets and capacities. As our results do not depend on these values, they are not affected by scaling all capacities and budgets by the smallest common denominator.

Note that finding a perfect minimum-weight matching in a bipartite graph $G = (U \cup V, E)$ is a special case of the min-cost flow problem [1].

2.2 Belief Propagation

For convenience, we describe the BP algorithm used by Bayati et al. [4]. For the details of the other versions of BP for the (perfect) maximum-weight b-matching problem and the min-cost flow problem we refer to the original works [2, 6, 10]. When necessary, we discuss the differences between the different versions of BP in Sections 4 and 5.

The BP algorithm used by Bayati et al. [4] is an iterative message-passing algorithm for computing maximum-weight matchings (MWM). Bayati et al. define their algorithm for complete bipartite graphs $G = (U \cup V, E)$ with $|U| = |V| = n$. In each iteration t, each node u_i sends a message vector $\vec{M}_{ij}^t = [\vec{m}_{ij}^t(1), \vec{m}_{ij}^t(2), \ldots, \vec{m}_{ij}^t(n)]$ to each of its neighbors v_j. The messages can be interpreted as how 'likely' the sending node thinks it is that the receiving node should be matched to a particular node in the MWM. The greater the value of the message $\vec{m}_{ij}^t(r)$, the more likely it is according to node u_i in iteration t that node v_j should be matched to node u_r. Similarly, each node v_j sends a message vector $\overleftarrow{M}_{ji}^t$ to each of its neighbors u_i. The messages are initialized as

$$\vec{m}_{ij}^0(r) = \begin{cases} w_{ij} & \text{if } r = i, \\ 0 & \text{otherwise} \end{cases} \qquad \text{and} \qquad \overleftarrow{m}_{ji}^0(r) = \begin{cases} w_{ij} & \text{if } r = j, \\ 0 & \text{otherwise.} \end{cases}$$

The messages in iterations $t \geq 1$ are computed from the messages in the previous iteration as follows:

$$\vec{m}_{ij}^t(r) = \begin{cases} w_{ij} + \sum_{k \neq j} \overleftarrow{m}_{ki}^{t-1}(j) & \text{if } r = i, \\ \max_{q \neq j} \left[w_{iq} + \sum_{k \neq j} \overleftarrow{m}_{ki}^{t-1}(q) \right] & \text{otherwise,} \end{cases} \quad \text{and}$$

$$\overleftarrow{m}_{ji}^t(r) = \begin{cases} w_{ij} + \sum_{k \neq i} \vec{m}_{kj}^{t-1}(i) & \text{if } r = j, \\ \max_{q \neq i} \left[w_{qj} + \sum_{k \neq i} \vec{m}_{kj}^{t-1}(q) \right] & \text{otherwise.} \end{cases}$$

The beliefs of nodes u_i and v_j in iteration t are defined as $b_{u_i}^t(r) = w_{ir} + \sum_k \overleftarrow{m}_{ki}^t(r)$ and $b_{v_j}^t(r) = w_{rj} + \sum_k \vec{m}_{kj}^t(r)$. The beliefs can be interpreted as

the 'likelihood' that a node should be matched to a particular neighbor. The greater the value of $b^t_{u_i}(j)$, the more likely it is that node u_i should be matched to node v_j. We denote the estimated MWM in iteration t by $\tilde{M}^t$. The estimated matching $\tilde{M}^t$ matches each node u_i to node v_j, where $j = \text{argmax}_{1 \leq r \leq n}\{b^t_{u_i}(r)\}$. Note that $\tilde{M}^t$ does not always define a matching, since multiple nodes may be matched to the same node. However, Bayati et al. [4] have shown that if the MWM is unique, then for t large enough, $\tilde{M}^t$ is a matching and equal to the MWM.

3 Isolation Lemma

3.1 Maximum-Weight Matchings

Beier and Vöcking [5] have considered a general scenario in which an arbitrary set $S \subseteq \{0,1\}^m$ of feasible solutions is given and to every $x = (x_1, \ldots, x_m) \in S$ a weight $w \cdot x = w_1 x_1 + \ldots + w_m x_m$ is assigned by a linear objective function. As in our model they assume that every coefficient w_i is drawn independently according to an adversarial density function $f_i : [0,1] \to [0,\phi]$ and they define δ as the difference in weight between the best and the second-best feasible solution from S, i.e., $\delta = w \cdot x^\star - w \cdot \hat{x}$ where $x^\star = \text{argmax}_{x \in S}\, w \cdot x$ and $\hat{x} = \text{argmax}_{x \in S \setminus \{x^\star\}}\, w \cdot x$. They prove a strong isolation lemma that, regardless of the adversarial choices of S and the density functions f_i, the probability of the event $\delta \leq \varepsilon$ is bounded from above by $2\varepsilon\phi m$ for any $\varepsilon \geq 0$.

If we choose S as the set of incidence vectors of all matchings or (perfect) b-matchings in a given graph, Beier and Vöcking's results yield for every $\varepsilon \geq 0$ an upper bound on the probability that the difference in weight δ between the best and second-best matching or the best and second-best (perfect) b-matching is at most ε. Combined with the results in Section 1 on the number of iterations needed by BP in terms of δ, this can immediately be used to obtain an upper tail bound on the number of iterations of the BP algorithm for these problems.

3.2 Min-Cost Flows

The situation for the min-cost flow problem is significantly more difficult because the set S of feasible integer flows cannot naturally be expressed with binary variables. If one introduces a variable for each edge corresponding to the flow on that edge, then $S \subseteq \{0,1,2,\ldots,u_{\max}\}^m$ where $u_{\max} = \max_{e \in E} u_e$. Röglin and Vöcking [9] have extended the isolation lemma to the setting of integer, instead of binary, vectors. However, their result is not strong enough for our purposes as it bounds the probability of the event $\delta \leq \varepsilon$ by $\varepsilon\phi m(u_{\max}+1)^2$ from above for any $\varepsilon \geq 0$. As this bound depends on $u_{\max}$ it would only lead to a pseudo-polynomial upper tail bound on the number of iterations of the BP algorithm when combined with the results of [6]. Our goal is, however, to obtain a polynomial tail bound that does not depend on the capacities. In the remainder of this section, we prove that the isolation lemma for integer programs [9] can

be significantly strengthened when structural properties of the min-cost flow problem are exploited.

In the following we consider the residual network for a flow f [1]. For each edge e_{ij} in the original network that has less flow than its capacity u_{ij}, we include an edge e_{ij} with capacity $u_{ij} - f_{ij}$ in the residual network. Similarly, for each edge e_{ij} that has flow greater than zero, we include the backwards edge e_{ji} with capacity f_{ij} in the residual network.

As all capacities and budgets are integers, there is always a min-cost flow that is integral. An additional property of our probabilistic model is that with probability one there do not exist two different integer flows with exactly the same costs. This follows directly from the fact that all costs are continuous random variables. Hence, without loss of generality we restrict our presentation in the following to the situation that the min-cost flow is unique.

In fact, Gamarnik et al. [6] have not used δ, the difference in cost between the best and second-best integer flow, to bound the number of iterations needed for BP to find the unique optimal solution of MCF, but they have used another quantity Δ. They have defined Δ as the length of the cheapest cycle in the residual network of the min-cost flow $f^\star$. Note that Δ is always non-negative. Otherwise, we could send one unit of flow along a cheapest cycle. This would result in a feasible integral flow with lower cost. With the same argument we can argue that Δ must be at least as large as δ because sending one unit of flow along a cheapest cycle results in a feasible integral flow different from $f^\star$ whose costs exceed the costs of $f^\star$ by exactly Δ. Hence any lower bound for δ is also a lower bound for Δ and so it suffices for our purposes to bound the probability of the event $\delta \leq \varepsilon$ from above.

The isolation lemma we prove is based on ideas that Gamarnik et al. [6, Theorem 8.1] have developed to prove that the optimal solution of a min-cost flow problem is unique with high probability if the costs are randomly drawn integers from a sufficiently large set. We provide a continuous counterpart of this lemma, where we bound the probability that the second-best integer flow is close in cost to the optimal integer flow.

Lemma 1. *The probability that the cost of the optimal and the second-best integer flow differs by at most $\varepsilon \geq 0$ is bounded from above by $2\varepsilon\phi m$.*

The isolation lemma (Lemma 1) together with the discussion about the relation between δ, the difference in cost between the best and second-best integer flow, and Δ, the length of the cheapest cycle in the residual network of the min-cost flow $f^\star$, immediately imply the following result.

Corollary 2. *For any $\varepsilon > 0$, we have $\mathbb{P}(\Delta \leq \varepsilon) \leq 2\varepsilon\phi m$.*

4 Upper Tail Bounds

4.1 Maximum-Weight Matching

We first consider the BP algorithm of Bayati et al. [4], which computes maximum-weight matchings in complete bipartite graphs G in $O(nw^\star/\delta)$ iterations

on all instances with a unique optimum. Here $w^\star$ denotes the weight of the heaviest edge and δ denotes the difference in weight between the best and the second-best matching. Even though it is assumed that G is a complete bipartite graph, this is not strictly necessary. If a non-complete graph is given, missing edges can just be interpreted as edges of weight 0.

With Beier and Vöcking's isolation lemma (see Section 3) we obtain the following tail bound for the number of iterations needed until convergence when computing maximum-weight perfect matchings in bipartite graphs using BP.

Theorem 3. *Let τ be the number of iterations until Bayati et al.'s BP [4] for maximum-weight perfect bipartite matching converges. Then $\mathbb{P}(\tau \geq t) = O(nm\phi/t)$.*

This tail bound is not strong enough to yield any bound on the expected running-time of BP for bipartite matchings. But it is strong enough to show that BP has smoothed polynomial running-time with respect to the relaxed definition adapted from average-case complexity [5], where it is required that the expectation of the running-time to some power $\alpha > 0$ is at most linear. However, a bound on the expected number of iterations is impossible, and the tail bound proved above is tight up to a factor of $O(m)$ (see Section 5).

As discussed in Section 1, BP has also been applied to finding maximum-weight (perfect) b-matchings in arbitrary graphs [2, 10]. The result is basically that BP converges to the optimal matching if the optimal solution of the relaxation of the corresponding linear program is unique and integral. The number of iterations needed until convergence depends again on "how unique" the optimal solution is. For Bayati et al.'s variant [2], the number of iterations until convergence depends on $1/\delta$, where δ is again the difference in weight between the best and the second-best matching. For Sanghavi et al.'s variant [10], the number of iterations until convergence depends on $1/c$, where c is the smallest rate by which the objective value will decrease if we move away from the optimal solution.

However, the technical problem in transferring the upper bound for bipartite graphs to arbitrary graphs is that the adversary can achieve that, with high probability or even with a probability of 1 (for larger ϕ), the optimal solution of the LP relaxation is not integral. Already in the average-case, i.e., for $\phi = 1$, where the adversary has no power at all, the optimal solution of the LP relaxation has some fractional variables with high probability.

Still, we can transfer the results for bipartite matching to both algorithms for arbitrary matching if we restrict the input graphs to be bipartite, since in this case the constraint matrix of the associated LP is totally unimodular.

Theorem 4. *Let τ be the number of iterations until Bayati et al.'s [2] or Sanghavi et al.'s [10] BP for general matching, restricted to bipartite graphs as input, converges. Then $\mathbb{P}(\tau \geq t) = O(nm\phi/t)$.*

Bayati et al. [2] and Sanghavi et al. [10] have also shown how to compute b-matchings with BP. If b is even, then the unique optimum to the LP relaxation

is integral. Thus, we circumvent the problem that the optimal solution might be fractional. Hence, following the same reasoning as above, the probability that BP for b-matching for even b runs for more than t iterations until convergence is also bounded by $O(mn\phi/t)$.

Furthermore, Bayati et al. [2, Section 4] have shown how to compute the optimal solution of the relaxation of the matching LP with *graph covers*. They obtain the same $O(n/\delta)$ bound for the number of iterations until convergence as for ordinary matching. However, since we are no longer talking about integer solutions, we cannot directly apply the isolation lemma of Beier and Vöcking [5]. To see that δ is still unlikely to be small in the same way (with a slightly worse constant), we can apply the isolation lemma of Röglin and Vöcking [9] since the matching polytope is half-integral. Thus, if we scale the right-hand side with a factor of 2, then we obtain a $0/1/2$ integer program. Because of this, we obtain the same $O(mn\phi/t)$ tail bound for the probability that the number of iterations until convergence exceeds t.

4.2 Min-Cost Flow

The bound for the probability that Δ is small (Corollary 2) together with the pseudo-polynomial bound of Gamarnik [6] yield a tail bound for the number of iterations that BP needs until convergence.

Theorem 5. *Let τ be the number of iterations until BP for min-cost flow [6] converges. Then $\mathbb{P}(\tau \geq t) = O(n^2 m\phi/t)$.*

5 Lower Tail Bounds

We show that the expected number of iterations necessary for convergence of BP for maximum-weight matching (MWM) is unbounded. To do this, we prove a lower tail bound on the number of iterations that matches the upper tail bound from Section 4 with respect to t. The lower bound holds even for a two by two complete bipartite graph with edge weights drawn independently and uniformly from the interval $[0, 1]$. In the following analysis, we consider the BP variant introduced by Bayati et al [4]. Our results can be extended to other versions of BP for matching and min-cost flow [2,6,10] in a straightforward way. We discuss these extensions at the end of this section.

5.1 Computation Tree

For proving the lower bounds, we need the notion of a *computation tree*, which we define analogously to Bayati et al. [4].

Let $G = (U \cup V, E)$ be a bipartite graph with $U = \{u_1, \ldots, u_n\}$ and $V = \{v_1, \ldots, v_n\}$. We denote the level-k *computation tree* with the root labeled $x \in U \cup V$ by $T^k(x)$. The tree $T^k(x)$ is a weighted rooted tree of height $k + 1$. The root node in $T^0(x)$ has label x, its degree is the degree of x in G, and its children

are labeled with the adjacent nodes of x in G. $T^{k+1}(x)$ is obtained recursively from $T^k(x)$ by attaching children to every leaf node in $T^k(x)$. Each child of a former leaf node labeled y is assigned one vertex adjacent to y in G as a label, but the label of the former leaf node's parent is not used. (Thus, the number of children is the degree of y minus 1.) Edges between nodes with label u_i and label v_j in the computation tree have a weight of w_{ij}.

We call a collection Λ of edges in the computation tree $T^k(x)$ a *T-matching* if no two edges of Λ are adjacent in $T^k(x)$ and each non-leaf node of $T^k(x)$ is the endpoint of exactly one edge from Λ. Leaves can be the endpoint of either one or zero edges from Λ. Let $t^k(u_i; r)$ be the weight of a maximum weight T-matching in $T^k(u_i)$ that uses the edge (u_i, v_r) at the root.

5.2 Average-Case Analysis

Consider the undirected weighted complete bipartite graph $K_{2,2} = (U \cup V, E)$, where $U = \{u_1, u_2\}$, $V = \{v_1, v_2\}$, and $(u_i, v_j) \in E$ for $1 \leq i, j \leq 2$. Each edge $(u_i, v_j) = e_{ij}$ has weight w_{ij} drawn independently and uniformly from $[0, 1]$. We define the event E_ε for $0 < \varepsilon \leq \frac{1}{8}$ as the event that $w_{11} \in \left[\frac{7}{8}, 1\right]$, $w_{12} \in \left(\frac{1}{2}, \frac{5}{8}\right]$, $w_{21} \in \left(\frac{5}{8}, \frac{3}{4}\right]$, and $w_{22} \in [w_{12} + w_{21} - w_{11} - \varepsilon, w_{12} + w_{21} - w_{11})$. Consider the two possible matchings $M_1 = \{e_{11}, e_{22}\}$ and $M_2 = \{e_{12}, e_{21}\}$. If event E_ε occurs, then the weight of M_2 is greater than the weight of M_1 and the weight differs by at most ε. In addition, w_{11} is greater than w_{12} and the weight differs by at least $1/4$. See Figure 1 for a graphical illustration.

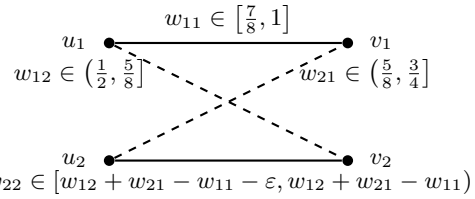

Fig. 1. If E_ε occurs, then the weight of the dashed matching $M_2 = \{e_{12}, e_{21}\}$ is greater than the weight of the solid matching $M_1 = \{e_{11}, e_{22}\}$ and the weight difference is at most ε. In addition w_{11} is greater than w_{12} and the weight difference is at least $\frac{1}{4}$.

Lemma 6. *The probability of event E_ε is $\varepsilon/8^3$.*

Lemma 7. *If event E_ε occurs, then the belief of node u_1 of $K_{2,2}$ at the end of the $4k$-th iteration is incorrect for all integers $k \leq \frac{1}{8\varepsilon} - 1$.*

By Lemma 6 and Lemma 7, we have a lower tail bound for the number of iterations that BP for MWM needs to converge for $K_{2,2}$.

Theorem 8. *The probability that BP for MWM needs at least t iterations to converge for $K_{2,2}$ with edge weights drawn independently and uniformly from $[0,1]$ is at least $\frac{1}{ct}$ for some constant $c > 0$.*

By using copies of $K_{2,2}$ we can extend the result of Theorem 8 to larger graphs.

Corollary 9. *There exist bipartite graphs on $n \geq 4$ nodes, where n is a multiple of 4, with edge weights drawn independently and uniformly from $[0,1]$, for which the probability that BP for MWM needs at least t iterations to converge is $\Omega\left(\frac{n}{t}\right)$ for $t \geq n/c'$ for some constant $c' > 0$.*

5.3 Smoothed Analysis

In this section, we consider complete bipartite graphs $K_{n,n}$ in the smoothed setting. We denote by $X \sim U[a,b]$ that the random variable X is uniformly distributed on interval $[a,b]$. In the following we assume that $\phi \geq 26$ and $n \geq 2$ and even. Similarly to the average case, we define the event E_ε^ϕ for $K_{2,2}$ and for $0 < \varepsilon \leq 1/\phi$ as the event that $w_{11} \in \left[1 - \frac{1}{\phi}, 1\right]$, $w_{12} \in \left(\frac{23}{26}, \frac{23}{26} + \frac{1}{\phi}\right]$, $w_{21} \in \left(\frac{23}{26}, \frac{23}{26} + \frac{1}{\phi}\right]$, and $w_{22} \in [w_{12} + w_{21} - w_{11} - \varepsilon, w_{12} + w_{21} - w_{11})$. Consider the two possible matchings $M_1 = \{e_{11}, e_{22}\}$ and $M_2 = \{e_{12}, e_{21}\}$. If event E_ε^ϕ occurs, then the weight of M_2 is greater than the weight of M_1 and the weight difference is at most ε. In addition w_{11} is greater than w_{12} and the weight difference is at least $\frac{3}{26} - \frac{2}{\phi}$. On this $K_{2,2}$, BP needs at least t rounds with a probability of $\Omega(\phi/t)$.

By taking $n/2$ copies of this $K_{2,2}$ and connecting all nodes in different parts of the bipartite graph by edges whose weights are drawn independently according to $U[0, \frac{1}{\phi}]$, we obtain a $K_{n,n}$ on which BP requires at least t rounds with a probability of $\Omega(\phi n/t)$.

Theorem 10. *There exist probability distributions on $[0,1]$ for the weights of the edges, whose densities are bounded by ϕ, such that the probability that BP for MWM needs at least t iterations to converge for $K_{n,n}$ is $\Omega(n\phi/t)$ for $t \geq n\phi/c$ for some constant $c > 0$.*

5.4 Other versions of BP

The results of this section also hold for other versions of belief propagation for minimum/maximum-weight (perfect) b-matching and min-cost flow [2,6,10] applied to the matching problem on bipartite graphs. The difference in the number of iterations until convergence differs no more than a constant factor. We omit the technical details but provide some comments on how the proofs need to be adjusted.

Some of the versions of BP consider minimum-weight perfect matching [2] or min-cost flow [6] instead of maximum-weight perfect matching. For these versions we get the same results if we have edge weights $\tilde{w}_e = 1 - w_e$ for all edges e.

For some of the versions of BP [6, 10] the root of the computation tree is an edge instead of a node . If we choose the root of this tree suitably, then we have that the difference in weight between the two matchings M_1 and M_2 of at most ε not only has to 'compensate' the weight difference $\Delta w(e_1, e_2)$ between an edge e_1 in M_1 and an edge e_2 in M_2, but the entire weight w_e of an edge e in M_1 or M_2. However, the probability distributions for the edge weights in Section 5 are chosen such that $\Delta w(e_1, e_2)$ and w_e do not differ more than a constant factor.

References

1. R. K. Ahuja, T. L. Magnanti, and J. B. Orlin. *Network flows: theory, algorithms, and applications*. Prentice-Hall, 1993.
2. M. Bayati, C. Borgs, J. Chayes, and R. Zecchina. Belief-propagation for weighted b-matching on arbitrary graphs and its relation to linear programs with integer solutions. *SIAM Journal on Discrete Mathematics*, 25(2):989–1011, 2011.
3. M. Bayati, A. Braunstein, and R. Zecchina. A rigorous analysis of the cavity equations for the minimum spanning tree. *Journal of Mathematical Physics*, 49(12):125206, 2008.
4. M. Bayati, D. Shah, and M. Sharma. Max-product for maximum weight matching: Convergence, correctness, and LP duality. *IEEE Transactions on Information Theory*, 54(3):1241–1251, 2008.
5. R. Beier and B. Vöcking. Typical properties of winners and losers in discrete optimization. *SIAM Journal in Computing*, 35(4):855–881, 2006.
6. D. Gamarnik, D. Shah, and Y. Wei. Belief propagation for min-cost network flow: Convergence & correctness. *Operations Research*, 60(2):410–428, 2012.
7. B. Manthey and H. Röglin. Smoothed analysis: Analysis of algorithms beyond worst case. *it – Information Technology*, 53(6):280–286, 2011.
8. J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann, 1988.
9. H. Röglin and B. Vöcking. Smoothed analysis of integer programming. *Mathematical Programming*, 110(1):21–56, 2007.
10. S. Sanghavi, D. M. Malioutov, and A. S. Willsky. Belief propagation and LP relaxation for weighted matching in general graphs. *IEEE Transactions on Information Theory*, 57(4):2203–2212, 2011.
11. S. Sanghavi, D. Shah, and A. S. Willsky. Message passing for maximum weight independent set. *IEEE Transactions on Information Theory*, 55(11):4822 –4834, 2009.
12. D. A. Spielman and S.-H. Teng. Smoothed analysis of algorithms: Why the simplex algorithm usually takes polynomial time. *Journal of the ACM*, 51(3):385–463, 2004.
13. D. A. Spielman and S.-H. Teng. Smoothed analysis: An attempt to explain the behavior of algorithms in practice. *Communications of the ACM*, 52(10):76–84, 2009.
14. J. S. Yedidia, W. T. Freeman, and Y. Weiss. Understanding belief propagation and its generalizations. In G. Lakemeyer and B. Nebel, editors, *Exploring Artificial Intelligence in the New Millennium*, chapter 8, pages 239–269. Morgan Kaufmann, 2003.