Separate but Equal: Equality in Belief Propagation for Single Cycle Graphs

Erel Cohen, Omer Lev and Roie Zivan
Ben Gurion University of the Negev
erelco@post.bgu.ac.il, omerlev@bgu.ac.il, zivanr@bgu.ac.il

Abstract

Belief propagation is a widely used incomplete optimization algorithm, whose main theoretical properties hold only under the assumptions that beliefs are not equal. Nevertheless, there is much evidence that equality between beliefs does occur. A method to overcome belief equality by using unary function-nodes is assumed to resolve the problem.

We focus on Min-sum, the belief propagation version for solving constraint optimization problems. We prove that on a single cycle graph, belief equality can be avoided only when the algorithm converges to the optimal solution. In any other case, the unary function methods will *not* prevent equality, rendering some existing results in need of reassessment. We differentiate between belief equality, which includes equal beliefs in a single message, and assignment equality, that prevents a coherent selection of assignments to variables. We show the necessary and satisfying conditions for both.

Introduction

The belief propagation algorithm (Pearl 1988; Yanover, Meltzer, and Weiss 2006) is an incomplete inference algorithm for solving problems that can be represented by graphical models. One major problem for which it can be applied is constraint optimization (Dechter 2003), a general model for centralized and distributed problem solving, which has a wide range of applications in artificial intelligence and multi agent systems (Ramchurn et al. 2010; Farinelli, Rogers, and Jennings 2014).

In belief propagation, each node in the graph acts as an agent within a distributed algorithm. In the standard synchronous version, this means that in each iteration of the algorithm it receives messages from all its neighboring nodes, performs computations, and sends messages to its neighbors. The agents maintain (and propagate) beliefs regarding the differences in the costs[1] which they will incur for assigning various possible value assignments to their variables.

Min-sum (often named Max-sum, e.g., Farinelli et al. (2008b); Rogers et al. (2011); Zivan et al. (2017); Cohen, Galiki, and Zivan (2020)) is a version of belief propagation that has drawn considerable attention (Ruozzi and Tatikonda 2013; Rogers et al. 2011; Chen et al. 2018), including being proposed for multi-agent applications such as sensor systems (Teacy et al. 2008; Stranders et al. 2009) and task allocation for rescue teams in disaster areas (Ramchurn et al. 2010). For convenience of presentation, we will focus on this version of belief propagation. However, the results that we present apply to other versions of belief propagation as well.

Belief propagation is known to converge to the optimal solution when solving problems represented by an acyclic graph. On problems represented by graphs that include cycles, the beliefs may fail to converge, and the resulting assignments that are considered optimal under those beliefs may be of low quality (Yanover, Meltzer, and Weiss 2006; Farinelli et al. 2008b; Zivan et al. 2017). This occurs because cyclic information propagation leads to computation of inaccurate and inconsistent information (Pearl 1988).

On graphs with a single cycle, it is known that belief propagation is not guaranteed to converge. However, when it does converge, the result is the optimal solution (Weiss 2000; Forney et al. 2001). Moreover, the conditions for convergence of belief propagation on a single cycle, are known: Forney et al. (2001) show the resemblance between the operation of the algorithm on a single cyclic graph and a chain of assignments (a route), which incurs costs with respect to the constraints along the chain. Every propagated belief is a result of a sum of costs, which are the result of the selection of value assignments in this route. Given enough iterations the route converges to a periodic traversal of the lowest cost sequence of assignments. The algorithm converges to the optimal solution if and only if this periodic route is consistent, i.e., nodes in the chain representing the same variable are not assigned different values. If it does include different value assignments to nodes representing the same variable, the algorithm will oscillate.

Recent work by Zivan, Lev, and Galiki (2020) tries to generalize the conditions for convergence of belief propagation, presented for single-cycle graphs (Forney et al. 2001; Weiss 2000), to the general case where graphs include multiple cycles. To do so, they proposed using a backtrack cost tree (BCT), which reveals the components that were summed in order to generate a propagated belief. They prove that after enough iterations, the bottom layers of all BCTs of a belief

[1]In this paper, without loss of generality, we will use costs and not utilities. Thus, the problem that the algorithm aims to solve is a minimization problem and violating a constraint incurs a finite cost.

included in a single message are identical. Furthermore, they show several other results regarding convergence in these cases.

There is a major setback in all the claims stated above: they are proven under the assumption that there is no equality between beliefs, i.e., the cost of different values that can be assigned to a single variable are never the same, so the best assignment at a particular stage is always clear. However, there is much theoretical and empirical evidence that when this assumption is not enforced, equal beliefs occur many times. Thus, there is a clear motivation to enforce no equality, in order for the results that were obtained under the no equality assumption to stand. Two attempts to enforce no equality were published: *Value Propagation*, commonly used within complete inference algorithms such as DPOP (Petcu and Faltings 2005), but also used for versions of Max-sum, e.g., Rogers et al. (2011); Zivan et al. (2017). However, this method is useful when the algorithm converges, and does not overcome pathologies that are triggered by belief equality during the algorithm's run, e.g., when solving graph coloring problems (Zivan et al. 2017). The second attempt to avoid belief equality was presented by Farinelli et al. (2008a). They proposed the use of unary constraints with randomly selected costs, which are smaller by orders of magnitude from the problem's constraints' costs. These are intended to reduce the probability of belief equality below a negligible threshold. We shall demonstrate in this paper that, while this method is effective when the algorithm converges to the optimal solution, e.g., when it is used for solving acyclic graphs, it does not prevent belief equality when there is as much as a single cycle in the graph, and the algorithm oscillates.

In this paper we extend the theory on belief propagation in general, and particularly on Min-sum, by making the following contributions:

1. We establish the conditions that lead to belief and assignment equality of Min-sum on graphs with a single cycle.
2. We prove that the unary constraint tie breaking method presented by Farinelli et al. (2008a) does not prevent ties even in a single cycle graph.

Our results are the first theoretical indication that ties cannot be avoided using the standard methods, when applying Min-sum to graphs that include cycles. This indicates that new theoretical analysis is needed to tackle such settings. Our empirical results demonstrate that when the algorithm does not converge to the optimal solution, the rate of occurrences of assignment equality is high.

The rest of the paper is organized as follows: We first provide background on graphical models, belief propagation and the existing theoretical knowledge regarding its convergence properties. Next, we present preliminaries followed by our theoretical results. Our empirical evaluation in which we examine how often is it that the algorithm does not converge and what is the rate of the existence of assignment equality in the cases where it does not converge. We conclude with our conclusions and suggestions for future work.

Background

In this section we present background on the graphical models to which our results apply, as well as the preliminaries of constraint optimization problems (COPs) and the version of belief propagation used for solving it – the Min-sum algorithm. We will also discuss the conditions for convergence on single cycle graphs, as presented in Forney et al. (2001).

Graphical Models

Graphical models, such as Bayesian networks or constraint networks, are a widely used representation framework for reasoning and solving optimization problems. The graph structure is used to capture dependencies between variables (Marinescu and Dechter 2009). Our work extends the theory established in Weiss (2000), which considered the maximum a posteriori (MAP) assignment, which is solved using the Max-product version of belief propagation. The relation between MAP and constraint optimization is well established (Marinescu and Dechter 2009; Farinelli et al. 2008a), and thus, results that consider Max-product for MAP apply to Max/Min-sum for solving constraint optimization problems, as well as the other way round (Ruozzi and Tatikonda 2013). Without loss of generality, we will focus on constraint optimization, since it is more common in AI literature. Our results apply to any version of problem represented by a graphical model and solved by belief propagation, as do the results of Weiss (2000).

Constraint Optimization

A COP is a tuple $\langle \mathcal{X}, \mathcal{D}, \mathcal{R} \rangle$. $\mathcal{X}$ is a finite set of variables $\{X_1, X_2, \ldots, X_m\}$. $\mathcal{D}$ is a set of domains $\{D_1, D_2, \ldots, D_m\}$ such that each domain D_i contains the finite set of values that can be assigned to variable X_i. We denote an assignment of value $x \in D_i$ to X_i by an ordered pair $\langle X_i, x \rangle$. $\mathcal{R}$ is a set of relations (constraints), and each constraint $R_j \in \mathcal{R}$ defines a non-negative *cost* for every possible value combination of a set of variables, and is of the form $R_j : D_{j_1} \times D_{j_2} \times \ldots \times D_{j_k} \rightarrow \mathbb{R}^+ \cup \{0\}$. A *binary constraint* refers to exactly two variables and is of the form $R_{ij} : D_i \times D_j \rightarrow \mathbb{R}^+ \cup \{0\}$.[2] For each binary constraint R_{ij} there is a corresponding cost table T_{ij} with dimensions $|D_i| \times |D_j|$ in which the cost in every entry $e_{x,y}$ is the cost incurred when X_i is assigned x and X_j is assigned y. A *binary COP* is a COP in which all constraints are binary. The *cost of a partial assignment* PA is the sum of all applicable constraints to PA over the value assignments in PA. A *complete assignment* (or a *solution*) is a partial assignment that includes all the COP's variables.An *optimal solution* is a complete assignment with minimal cost. For simplicity, we concentrate only on binary COPs.

Min-Sum Belief Propagation

Min-sum operates on a *factor-graph*, which is a bipartite graph in which the nodes represent variables and constraints

[2]We say that a variable is *involved* in a constraint if it is one of the variables the constraint refers to.

(Kschischang, Frey, and Loeliger 2001). Each variable-node, representing a variable of the original COP, is connected to all function-nodes which represent the constraints that involve it. Similarly, a function-node is connected to all variable-nodes which are involved in the constraint it represents. Variable-nodes and function-nodes take an active role in Min-sum, i.e., they can send and receive messages, and can perform computation. Min-sum can work in distributed settings, where each node's role is performed by an autonomous agent, or in centralized settings in which the role of all nodes is managed by a single entity.

A message sent to – or from – variable-node X (for simplicity, we use the same notation for a variable and the variable-node representing it) is a vector of size $|D_X|$ including a cost for each value in D_X. Before the first iteration, all nodes assume that all messages they previously received (in iteration 0) include vectors of zeros. A message sent from a variable-node X to a function-node F in iteration $i \geq 1$ is formalized as follows:

$$Q^i_{X\to F} = \sum_{F' \in F_X, F' \neq F} R^{i-1}_{F' \to X} - \alpha$$

where $Q^i_{X\to F}$ is the message variable-node X intends to send to function-node F in iteration i, F_X is the set of function-node neighbors of variable-node X and $R^{i-1}_{F'\to X}$ is the message sent to variable-node X by function-node F' in iteration $i-1$. α is a constant that is reduced from all costs included in the message (i.e., for each $x \in D_X$) in order to prevent the costs carried by messages from growing arbitrarily large during the algorithm's run.

A message $R^i_{F\to X}$ sent from a function-node F to a variable-node X in iteration i, includes for each value $x \in D_X$:

$$R^i_{F\to X} = min_{PA_{-X}} cost(\langle X, x\rangle, PA_{-X})$$

where PA_{-X} is a possible combination of value assignments to variables involved in F not including X. The term $cost(\langle X, x\rangle, PA_{-X})$ represents the cost of a partial assignment $a = \{\langle X, x\rangle, PA_{-X}\}$, which is:

$$f(a) + \sum_{X' \in X_F, X' \neq X, \langle X', x'\rangle \in a} (Q^{i-1}_{X'\to F})_{x'}$$

where $f(a)$ is the original cost in the constraint represented by F for the partial assignment a, X_F is the set of variable-node neighbors of F, and $(Q^{i-1}_{X'\to F})_{x'}$ is the cost that was received in the message sent from variable-node X' in iteration $i-1$, for the value x' that is assigned to X' in a. X selects its value assignment $\hat{x} \in D_X$ following iteration k as follows:

$$\hat{x} = \arg\min_{x \in D_X} \sum_{F \in F_X} (R^k_{F\to X})_x$$

Split Constraint Factor Graphs

A constraint in a factor graph is commonly represented by a single function-node. However, it is possible to split a constraint and represent it by two (or more) function-nodes (Ruozzi and Tatikonda 2013; Cohen, Galiki, and Zivan 2020). Such *Split Constraint Factor Graphs* (SCFGs) are used as an enhancement method for Min-sum. This is achieved by having each constraint that was represented by a single function-node in the original factor graph, represented by two function-nodes. The SCFG is equivalent to the original factor graph, if the sum of the cost tables of the two function-nodes representing each constraint in the SCFG is equal to the cost table of the single function-node representing the same constraint in the original factor graph. By tuning the similarity between the two function-nodes representing the same constraint one can determine the level of asymmetry in the SCFG (Cohen, Galiki, and Zivan 2020).

Single Cycle Factor Graphs

For a single cycle factor graph, we know that if belief propagation converges, it is to the optimal solution (Forney et al. 2001; Weiss 2000). Moreover, when the algorithm does not converge – it periodically changes its set of assignments. Explaining this behavior, Forney et al. (2001) show the similarity of the performance of the algorithm on a cycle to its performance on a tree, whose nodes are similar to the nodes in the cycle, but whose length is equal to the number of iterations performed by the algorithm. One can consider a sequence of messages starting at the first node of the chain and heading towards its other end. Each message carries beliefs accumulated from costs added by function-nodes. Each function-node adds a cost to each belief, which is the constraint value of a pair of value assignments to its neighboring variable-nodes. Each such sequence of cost accumulation (route) must at some point become periodic, and the minimal belief would be generated by the minimal periodic route. If this periodic route is consistent, i.e., the set of assignments implied by the costs in it contain a single value assignment for each variable, the algorithm converges and the implied assignment is the optimal solution; otherwise, it does not (Forney et al. 2001). Our results demonstrate that there are cases where ties cannot be avoided. Specifically, when the algorithm oscillates, under some conditions, the minimal repeated route that the algorithm follows includes more than one value in each variable's domains and the beliefs for these values are become equal periodically. This phenomenon cannot be avoided using the unary constraint method proposed in Farinelli et al. (2008b).

Preliminaries

Our analysis will include insights regarding the construction of beliefs from costs incurred by constraints on a single cycle graph. Let us address the problem in which there are n variable nodes in a single cycle graph. We shall mark the state of the algorithm at time t (that is, after t iterations) as $\vec{s}^t$. This state includes the value assignments selected by all variable-nodes in the graph at time t. The value assignments are selected according to the R messages sent to the variable-nodes by their function-node neighbors.

For every pair of constrained variables, X_i and X_j, for each $x \in D_i$, $x' \in D_j$, we will denote the cost incurred according to the constraint for assigning x to X_i and x' to X_j as $C_{(X_i=x, X_j=x')}$. This is the cost specified by the corresponding cost entry in the cost table held by the function-node representing the constraint between X_i and X_j.

Forney et al. (2001) proved that when belief propagation is applied to a single cycle graph, there is a time t_0 and a positive integer v such that for any $t \geq t_0$, the state $s^t = s^{t+v}$, i.e., starting at time t_0 the algorithm produces the same state every v iterations. Forney et al. (2001) further proves that if the algorithm converges, i.e., $v = 1$, then the single repeated state is the optimal solution. Intuitively, its optimality follows from the optimality of belief propagation on trees in general and specifically on chains. If we select an arbitrary long chain representing the solving process of belief propagation on the single cycle graph, then it would include the assignment that is derived from the repeated set periodically for any number of times that we chose. Thus, if this assignment was not optimal, that would mean that there is a different assignment that is an optimal solution for this chain, and thus, the optimality of the algorithm on chain structured graphs would be contradicted.

Definition 1. *A Backtrack Cost Tree (BCT) for a belief b^k is a hierarchical structure that specifies the constraint costs that were summed in order to generate b^k in iteration k. b^k generated by the sum of costs (beliefs) that were sent in R messages. Thus, if b^k is a belief sent in message R in iteration k, then for $k > 1$, it is the sum of beliefs that were sent in messages in iteration $k-2$ and an additional cost $C_{(X_i=x, X_j=x')}$ that is an entry in the cost table of the function-node sending the R message that includes b^k. Similarly and recursively, each belief b^{k-2} that was sent in a message in iteration $k-2$ and was used to generate b^k, was generated by summing beliefs sent in iteration $k-4$ (assuming $k > 3$) and an entry in the cost table of the function-node sending the message including b^{k-2}. This backtrack process continues until we reach iteration 1 and all the costs that were summed in order to generate b^k are revealed.*

In a single cycle graph, each BCT is a chain. We highlight the table cost entries that are summed in a BCT by using the next definition.

Definition 2. *The route of a $BCT(b^k)$ in a single cycle graph be the ordered list of table cost entries $C_{(X_i=x, X_j=x')}$ that are summed by the algorithm in order to generate belief b^k. We will denote the route r of $BCT(b^k)$ by r_{b^k}.*

We will say that the route *visits* a table cost entry, when this entry is added to the route.

For each route there is a corresponding assignment that is implied by it. That is, if the k'th cost in the route is $C_{(X_i=x, X_j=x')}$, this implies that the k and $k+1$ value assignments in the implied assignment are $X_i = x$ and $X_j = x'$ respectively.

Definition 3. *Let the route of the BCT of the minimal belief in a message, be the* **minimal route** *and its corresponding assignment be the* **minimal assignment**.

Following Forney et al. (2001) we know that there is a time t_0 such that for any time $t > t_0$, the minimal route includes the minimal normalized sequence of entries that repeats itself periodically. When this list's periodic interval is the size of the function nodes in the graph, this means that the algorithm has converged and the minimal assignment is the optimal solution (since this means the assignment of every variable is always the same, whenever we "visit" it in our sequence).

We will refer to the set of cost table entries $C_{(X_i=x, X_j=x')}$ that are included in the minimal route following t_0 as the set M.

The following observation will assist in establishing our results.

Observation 1. *In any given route r, for every two consecutive cost table entries in r, $C_{(X_i=x, X_j=x')}$ and $C_{(X_j=y, X_s=y')}$, $x' = y$.*

This observation is inferred directly from the definition of messages above, discussing how the R messages in Min-sum are created. See example in Figure 2.

The implication of this observation is that the number of possible routes of length m, assuming the variables' domain size is d, is d^n and not d^{2n}. It also helps to follow the flow of the algorithm and – given some route – to predict what the next table cost entry in this route will be.

Conditions for Belief and Value Assignment Equality

In order to study the cause for the belief equality phenomenon, i.e., messages that include an identical cost for at least two value assignments, we analyze the components from which these beliefs were composed. That is, we investigate what entries in the constraints' cost tables are summed in the process that generates the equal beliefs. Since we can use the unary constraint method proposed in Farinelli et al. (2008b), we assume that belief equality is possible if and only if the components of the routes of a BCT are identical. In any other case – when the components being added up are not identical – we assume there is a difference in the outcome, and therefore there is some a minimal belief in each message.

Observation 2. *In the minimal route, following t_0, the order of the route elements (table cost entry elements) is fixed.*

This observation is straightforward from each entry being equivalent to a value $C_{X_i=x, X_j=y}$, and each element being sufficiently different such that a particular value of belief can only be produced by a single set of table cost entry values (and, as noted above, Forney et al. (2001) shows that following t_0, there is a fixed periodic minimal route).

Lemma 1. *When the algorithm does not converge, then there must be at least one function-node F with two cost table entries in M.*

Proof. If this is not true then there is a single value assignment for each variable and that means that the algorithm does converge. □

Proposition 1. *When the algorithm does not converge, and assuming there are no tied beliefs, then any two entries in M, which belong to the cost table of the same function-node* *cannot be on the same row or column of the cost table, i.e., cannot imply the same value assignments.* [3]

Proof. Assume that the proposition is false. For a function-node $F_{i,j}$ representing the constraint between X_i and X_j, consider the message that is sent from $F_{i,j}$ to X_j. In this message there is a minimal belief which corresponds to the assignment $X_i = x$. Thus, the relevant entries to extend this route are the entries in the vector $C_{(X_i=x, X_j=\cdot)}$, from which only one is minimal. Thus, if there are more than one $C_{(X_i=x, X_j=\cdot)}$ entries included in M, this contradicts the the minimal route property.

Since we have the algorithm working in both directions, a similar claim rules out two entries with the same value assigned to X_j. □

Proposition 2. *In the periodic cycle that begins after t_0, in every cycle of size vn, each table cost entry that is included in M is visited exactly once.*

Proof. Assume the proposition does not hold. Thus, there must be an entry e in some function table (e.g., $C_{X_i=x, X_j=y}$), that the route includes more than one entry that come right after it, i.e., that there are two entries e' and e'' such that, in the route, the entry that comes right after e will e' or e'' alternately (e.g., $e' = C_{X_j=j, X_k=z}$; $e'' = C_{X_j=j, X_k=z'}$). However, this will mean that e' and e'' are both included in M and have the same value assignment for one of their variables. This contradicts Proposition 1. □

An immediate corollary from proposition 2 is that the number of entries that are visited in each function-node cost table in a periodic cycle of the minimal route of size v is exactly v.

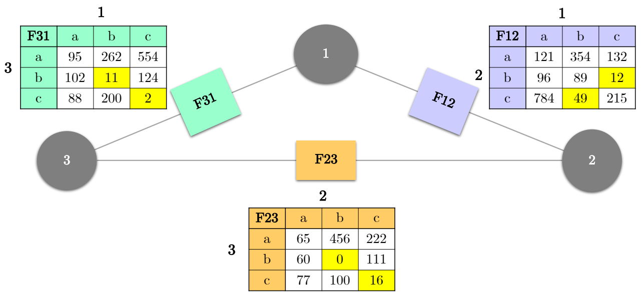
Immediate Convergence In the following part of our discussion we will assume that $t_0 = 0$, i.e., that the algorithm immediately converges to a minimal route.

Theorem 1. *When the algorithm runs on a single cycle graph and does not converge (i.e., $v > 1$), and $t_0 = 0$, for any natural number k, after kvn iterations the beliefs sent to all value assignments within the minimal assignment are equal.*

Proof. From Proposition 2 we know that following t_0, in every vn iterations, the minimal route visits each of the entries in M exactly once. Since $v > 1$, we know that multiple values of M are visited for each function node which are not in the same line or row, i.e., each time a different value is minimal. Thus, if $t_0 = 0$ then in the first vn iterations each of the entries in M will be visited exactly once by BCTs that follow the minimal route every vn iterations. This means that all the possible values in the function node which were minimal at some point in the route (i.e., were part of M) to be composed of the same elements – reaching equality. □

An immediate corollary from Theorem 1 is that when $v > 1$ and $t_0 = 0$, for any natural number k, after kvn iterations

[3]As Figure 1 shows, the yellow cells – which are the cells in M in every function-node – are not in the same row or column with each other.



F31	a	b	c
a	95	262	554
b	102	11	124
c	88	200	2

F12	a	b	c
a	121	354	132
b	96	89	12
c	784	49	215

F23	a	b	c
a	65	456	222
b	60	0	111
c	77	100	16

Figure 1: A cycle with 3 nodes, where each variable has a domain of size 3.

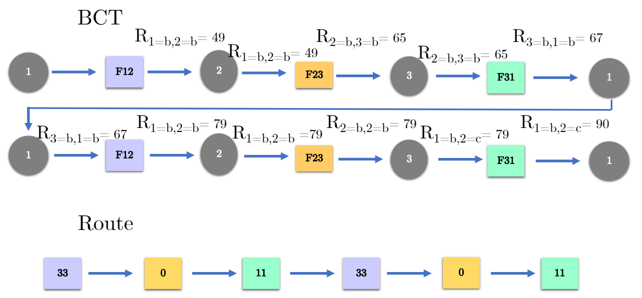


Figure 2: BCT and Route of the belief sent to variable-node 1 for value b, according to the graph in Figure 1.

each variable-node will have assignment equality between v values in its domain.

Example 1. *In the following example we will use the rather simple case where we have a graph with three nodes, with each variable each with a domain of size 3, i.e., $n = 3$ and $d = 3$. Figure 1 presents the instance of the cycle graph that we will discuss, including the function-node cost tables.*

We highlight in yellow the entries in the cost tables that are included in the set M, i.e. visited repeatedly in the minimal route, following time t_0. In this example, each of the highlighted costs is the lowest in its row and its column, Thus, these will be selected right away by the algorithm and therefore $t_0 = 0$.

As proven in Theorem 1, the cost of all beliefs corresponding to values in the minimal route, which are the coordinates of all the entries in M will be equal after visiting each of the members in M once. Since we have $2n$ nodes in the single cycle factor-graph, the number of iterations required to visit the six members in M is 12.

Figure 2 presents the BCT and corresponding route for the belief sent at the 12'th iteration to variable-node 1 for its value b. The BCT accumulates the members of M in the order that is displayed in the route. During the following 12 iterations each of the entries in the route will be visited again in the same order. The same entries (only in a different order) will be also visited by the BCT and route of the belief sent to variable-node 1 for its value c. Thus, every 12 iterations we will have belief equality.

If we replace the entry in F_{12} of the cost for $1 = b, 2 = b$

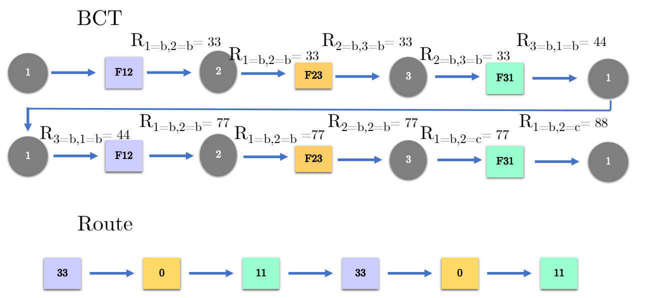


Figure 3: BCT and Route of the belief sent to variable-node 1 for value b, after changing the graph in Figure 1, so that the entry $\langle b, b\rangle$ of F_{12} is equal to 33.

(that is currently 89) to a number smaller than 34 the solution $1 = b, 2 = b, 3 = b$ is optimal and the set M includes the entries $\langle b, b\rangle$ in each of the function-nodes. In this case there is no belief equality. Figure 3 presents the BCT and minimal route that Min-sum converges to after replacing the entry $\langle b, b\rangle$ in F_{12} from 89 to 33. Notice that the accumulated cost of the route in this case is 88, which is smaller than 90. If we replace this entry by 42 then we have a "tail" (defined and detailed below), since in the first message sent from F_{12} to variable-node 1, the entry $2 = b, 1 = b$ will be selected although it is not a part of M. This tail prevents belief equality, but does not prevent assignment equality since the same entry is selected while generating the tail in the other direction, in the second message sent from F_{12} to variable-node 2.

Non-Immediate Convergence We now turn to discuss the case where $t_0 > 0$, i.e., the algorithm does not start right away to follow a periodic path, but will take several iterations to reach it (Weiss 2000; Forney et al. 2001).

Definition 4. *For each route r, we will call the part that comes before t_0 the* tail *of the route and we will denote it by $tail_r$.*

When a variable-node in a single cycle graph selects a value assignment for its variable, it sums the beliefs received in the R messages sent to it in the last iteration. Thus, at every time $t \in \mathbb{N}$, for each value x in its domain it receives two beliefs, and for each of them there is a route. One route is accumulated in messages arriving in the clockwise direction which we denote by r_x^{cw}, and the other is accumulated in a counterclockwise direction, denoted by r_x^{ccw}. These routes have tails which we will denote by $tail_x^{cw}$ and $tail_x^{ccw}$ respectively.

We will show that when the algorithm does not converge and does not start at t_0, the existence of belief equality and assignment equality is dependent on the similarity between these tails.

Theorem 2. *When Min-sum is applied to a single cycle graph, belief equality in some message will occur if and only if all the following conditions hold:*

1. *$t - t_0$, i.e., the number of iterations since convergence to a periodic pattern, is kvn, for any $k \in \mathbb{N}$.*
2. *$v > 1$.*
3. *There is a variable X_i such for some two values x, x' in its domain which belong to the minimal assignment, receive beliefs through routs r_x and $r_{x'}$ which have tails ($tail_{r_x}$ and $tail_{r_{x'}}$, respectively) composed of the same entries and therefore their sum is equal.*

Proof. Assume the three conditions hold. Then, according to Theorem 1 the routes of the beliefs corresponding to the value assignments in the minimal assignment following t_0 are equal. Thus, if their tails are equal as well, the theorem holds.

Assume that the beliefs in a message belonging to the minimal assignment are equal, then according to our assumption that constraint values are different (due to unary constraints), the value assignments are equal only if their routes include exactly the same entries. Following t_0 this can happen only when $v > 1$ and every kvn iterations (since for each of the routes the entry it visits following t_0 is different). Obviously, in order for the complete routes to be equal to each other, i.e., to be composed of the same components, the third condition must hold as well. □

Similarly to the conditions of belief equality established above, the following theorem establishes the conditions for assignment equality.

Theorem 3. *Assignment equality occurs if and only if the following three condisions hold:*

1. *$t - t_0$, i.e., the number of iterations since convergence to a periodic pattern, is kvn, for any $k \in \mathbb{N}$.*
2. *$v > 1$.*
3. *There is a variable X_i such for some two values x, x' in its domain which belong to the minimal assignment, the following equation holds:*

$$\sum_{e \in tail_x^{cw}} e + \sum_{e \in tail_x^{ccw}} e = \sum_{e \in tail_{x'}^{ccw}} e + \sum_{e \in tail_{x'}^{cw}} e$$

Proof. Assume the three conditions hold. Then, according to Theorem 1 the routes of all beliefs intended for the value assignments in the minimal assignment following t_0 are equal. Thus, if the sum of the tails of the two routes for each value assignment is equal to the sum of the tail of all other value assignments within the minimal assignment, then the theorem holds.

Assume that the sum of beliefs received by a variable-node, for each of the possible value assignments that belong to the minimal assignment are equal, then according to our assumption, they are equal only if the union of their routes include exactly the same entries. Following t_0 this can happen only when $v > 1$ and every kvn iterations (for similar reasons to the ones described in the proof for Theorem 2). In order to have assignment equality we must have that the sum of the tails of the routes from which the beliefs corresponding to one value assignment in the domain, which belongs to the minimal assignment, will be equal to the sum of the tails of the routes of the beliefs of every other value assignment in the domain that belongs to the minimal assignment. This is condition three. □

The significance of the theoretical properties we establish is in the understanding that the element that determines whether Min-sum will generate belief or assignment equality, is a *property of the problem*, and that methods that the unary constraint method cannot overcome this property. This is because the equalities are generated by the same constraint values being accumulated in different routes by the algorithm. Thus, as long as there is no alternative method for avoiding ties during the run of the algorithm, one cannot assume that on graphs that include cycles, equalities can be avoided.

Experimental Evaluation

Following our theoretical proofs, we wanted to gain a perspective on how common is assignment equality in Min-sum when applied to a single cycle factor-graph. To do so, we created simulations in which we varied the size of the cycle and the size of the domain, and generated random instances in which each entry in the cost table of a constraint was a natural number selected uniformly from the range $[1, 100]$. To simulate Farinelli et al. (2008a)'s method of avoiding equalities, we assigned random unary constraints, selected uniformly from the range $[10^{-8}, 10^{-4}]$. Since our claims mainly deals with cases which do not converge, for each pair of domain size and cycle size (the number of variable-nodes in the cycle) we generated random instances, solving them using Min-sum, until we had 100 instances on which Min-sum did not converge to an optimal solution. Table 1 presents for each such combination the number of instances generated in order to find 100 instances on which Min-sum did not converge, and in Table 2 we can see the number of times among these 100 instances that Min-sum generated assignment equalities. The results presented in Table 1 demonstrates clear trends: the probability that the algorithm will converge decreases as the cycle size grows; and the probability of convergence grows when the domain size grows.

The results presented in Table 2 are much more stable. It is clear that among the instances on which Max-sum does not converge, the rate of instances on which assignment eqaulities was generated was high. For the smallest cycles with only three variable-nodes, we see some relation between the size of the domain and the number of instances in which Min-sum generated assignment equality. Surprisingly, we see that for larger cycles, this rate is less dependent on the size of the domains. We also ran this test for cycles with 10 variable nodes and domain size of 10 and received on 91% of the instances that did not converge, assignment equality. Our conclusion is that in most cases, the tails generated before the algorithm starts its periodical path (i.e., before t_0) are symmetric.

		Cycle size		
		3	**4**	**5**
Domain size	**2**	2240	5297	10000+
	3	837	1424	2205
	4	734	762	1447
	5	414	535	912

Table 1: Number of instances solved until 100 instances on which Min-sum does not converge were found.

		Cycle size		
		3	**4**	**5**
Domain size	**2**	96	98	96
	3	97	96	96
	4	81	94	99
	5	84	92	96

Table 2: Number of instances among the 100 that did not converge in which assignment equality was generated.

Conclusion

Belief propagation is a well known and widely used algorithm for solving combinatorial optimization problems that can be represented by a graphical model. The theoretical knowledge regarding this algorithm is limited to specific graph structures such as acyclic graphs and graphs with a single cycle. Yet, even this limited knowledge is based on the assumption that ties between beliefs do not exist. In order to avoid belief equality, the unary constraints method, that assigns every possible value assignment with a randomly selected very small cost, was proposed by Farinelli et al. (2008b). This method intends to reduce the probability of ties beneath a negligible threshold.

When the algorithm converges, such as in the case where the algorithm solves a tree-structured graph, this does indeed work. However, we prove that even when graphs include a single cycle, this method cannot prevent belief and assignment equalities. Thus, *ties cannot be avoided*, and our results establish the conditions for such equalities in a graph with a single cycle. Moreover, our simulations indicate that these equalities are a key reason for lack of convergence. We suspect that this phenomenon occurs in more elaborate graphs as well, with more than one cycle. This understanding implies that some of the theoretical knowledge that was based on the assumption that ties can be avoided in belief propagation, needs to be reevaluated.

Thus, this work opens up a wide field of research. Firstly, existing results must be re-examined to see how the existence of ties affects them. Second, additional methods for avoiding ties should be suggested and can be considered novel. In our future research, we plan to investigate whether the results we presented in this paper also apply when methods that are known to immensely improve the performance of Min-sum on graphs with multiple cycles, such as damping and splitting, are used.

Acknowledgments

This work was supported, in part, by Israel Science Fund grants 1965/20 and 3152/20.

References

Chen, Z.; Deng, Y.; Wu, T.; and He, Z. 2018. A class of iterative refined Max-sum algorithms via non-consecutive value propagation strategies. *Autonomous Agents and Multi Agent Systems*, 32(6): 822–860.

Cohen, L.; Galiki, R.; and Zivan, R. 2020. Governing convergence of Max-sum on DCOPs through damping and splitting. *Artif. Intell.*, 279.

Dechter, R. 2003. *Constraint Processing*. Morgan Kaufman.

Farinelli, A.; Rogers, A.; and Jennings, N. R. 2014. Agent-based decentralised coordination for sensor networks using the max-sum algorithm. *Autonomous Agents and Multi-Agent Systems*, 28(3): 337–380.

Farinelli, A.; Rogers, A.; Petcu, A.; and Jennings, N. 2008a. Decentralised coordination of low-power embedded devices using the max-sum algorithm. In *Proceedings of the 7th International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 2*, 639–646. International Foundation for Autonomous Agents and Multiagent Systems.

Farinelli, A.; Rogers, A.; Petcu, A.; and Jennings, N. R. 2008b. Decentralized Coordination of Low-Power Embedded Devices Using the Max-Sum Algorithm. In *AAMAS*, 639–646.

Forney, G. D.; Kschischang, F. R.; Marcus, B.; and Tuncel, S. 2001. Iterative decoding of tail-biting trellises and connections with symbolic dynamics. In Marcus, B.; and Rosenthal, J., eds., *Codes, Systems, and Graphical Models*, 239–264. Springer.

Kschischang, F. R.; Frey, B. J.; and Loeliger, H. A. 2001. Factor Graphs and the Sum-Product Algorithm. *IEEE Transactions on Information Theory*, 47:2: 181–208.

Marinescu, R.; and Dechter, R. 2009. AND/OR Branch-and-Bound search for combinatorial optimization in graphical models. *Artif. Intell.*, 173(16-17): 1457–1491.

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

Petcu, A.; and Faltings, B. 2005. A Scalable Method for Multiagent Constraint Optimization. In *IJCAI*, 266–271.

Ramchurn, S. D.; Farinelli, A.; Macarthur, K. S.; and Jennings, N. R. 2010. Decentralized Coordination in RoboCup Rescue. *Computer J.*, 53(9): 1447–1461.

Rogers, A.; Farinelli, A.; Stranders, R.; and Jennings, N. R. 2011. Bounded Approximate Decentralized Coordination via the Max-sum Algorithm. *Artificial Intelligence*, 175(2): 730–759.

Ruozzi, N.; and Tatikonda, S. 2013. Message-Passing Algorithms: Reparameterizations and Splittings. *IEEE Trans. Information Theory*, 59(9): 5860–5881.

Stranders, R.; Farinelli, A.; Rogers, A.; and Jennings, N. R. 2009. Decentralised Coordination of Mobile Sensors Using the Max-Sum Algorithm. In *IJCAI 2009, Proceedings of the 21st International Joint Conference on Artificial Intelligence, Pasadena, California, USA, July 11-17, 2009*, 299–304.

Teacy, W. T. L.; Farinelli, A.; Grabham, N. J.; Padhy, P.; Rogers, A.; and Jennings, N. R. 2008. Max-sum decentralized Coordination for Sensor Systems. In *AAMAS*, 1697–1698.

Weiss, Y. 2000. Correctness of Local Probability Propagation in Graphical Models with Loops. *Neural Computation*, 12(1): 1–41.

Yanover, C.; Meltzer, T.; and Weiss, Y. 2006. Linear Programming Relaxations and Belief Propagation - An Empirical Study. *Journal of Machine Learning Research*, 7: 1887–1907.

Zivan, R.; Lev, O.; and Galiki, R. 2020. Beyond Trees: Analysis and Convergence of Belief Propagation in Graphs with Multiple Cycles. In *AAAI*, 7333–7340. IL.

Zivan, R.; Parash, T.; Cohen, L.; Peled, H.; and Okamoto, S. 2017. Balancing exploration and exploitation in incomplete Min/Max-sum inference for distributed constraint optimization. *Autonomous Agents and Multi-Agent Systems*, 31(5): 1165–1207.