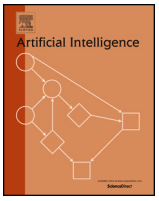
Artificial Intelligence 338 (2025) 104243

Contents lists available at ScienceDirect

Artificial Intelligence

journal homepage: www.elsevier.com/locate/artint



# Separate but equal: Equality in belief propagation for single-cycle graphs ☆

Erel Cohen, Ben Rachmut, Omer Lev, Roie Zivan *

*Department of Industrial Engineering and Management, Ben-Gurion University of the Negev, Beersheba, Israel*

ARTICLE INFO

*Keywords:*
Belief propagation
Distributed constraints
Distributed problem solving

ABSTRACT

Belief propagation is a widely used, incomplete optimization algorithm whose main theoretical properties hold only under the assumption that beliefs are not equal. Nevertheless, there is substantial evidence to suggest that equality between beliefs does occur. A published method to overcome belief equality, which is based on the use of unary function-nodes, is commonly assumed to resolve the problem.

In this study, we focus on min-sum, the version of belief propagation that is used to solve constraint optimization problems. We prove that for the case of a single-cycle graph, belief equality can only be avoided when the algorithm converges to the optimal solution. Under any other circumstances, the unary function method will *not* prevent equality, indicating that some of the existing results presented in the literature are in need of reassessment. We differentiate between belief equality, which refers to equal beliefs in a single message, and assignment equality, which prevents the coherent assignment of values to the variables, and we provide conditions for both.

## 1. Introduction

The belief propagation algorithm [11,21] is an incomplete inference algorithm for solving problems that can be represented by graphical models. One major domain to which it can be applied is constraint optimization [3], a general model for centralized and distributed problem-solving, which has a wide range of applications in artificial intelligence and multi-agent systems [13,5].

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

Min-sum is a version of belief propagation that has drawn considerable attention [15,14,1], including being proposed for multi-agent applications such as sensor systems [18,16] and task allocation for rescue teams in disaster areas [13]. When solving maximization problems, the algorithm is referred to as *max-sum*. This name is commonly used in the distributed constraint optimization community even when the problem solved is a minimization problem (see, e.g., Zivan et al. [22], Cohen et al. [2]). For convenience

☆ This paper is an extension of our AAAI 2023 paper.

* Corresponding author.
*E-mail addresses:* erelco@post.bgu.ac.il (E. Cohen), rachmut@post.bgu.ac.il (B. Rachmut), omerlev@bgu.ac.il (O. Lev), zivanr@bgu.ac.il (R. Zivan).

[1] In this paper, without loss of generality, we consider 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.

https://doi.org/10.1016/j.artint.2024.104243
Received 10 May 2023; Received in revised form 22 October 2024; Accepted 23 October 2024

Available online 8 November 2024

of presentation, we will focus solely on the min-sum 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 [6,14]. When problems are 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 [21,4,22]. This occurs because cyclic information propagation leads to computation of inaccurate and inconsistent information [11].

When solving problems represented by 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 [19,6]. Moreover, the conditions for convergence of belief propagation on a single cycle are known: Forney et al. [6] show the resemblance between the operations of the algorithm on a single-cycle graph and its operations when solving a chain structured graph. Every propagated belief corresponds to a sum of costs, where each cost is the result of the assignment of a set of values. After a sufficient number of iterations, the algorithm starts a periodic traversal of the lowest-cost sequence of assignments (a minimal route). The algorithm converges to the optimal solution if and only if this periodic route is consistent, i.e., if nodes in the chain structured graph that represent the same variable are assigned consistent values. If, on the other hand, the route includes different values assigned to nodes representing the same variable, the algorithm will oscillate.

Recent work by Zivan et al. [23] proposed the use of a backtrack cost tree (BCT), which reveals the cost components that were summed in order to generate a propagated belief. They prove (among other results) that after a sufficient number of iterations, the bottom layers of all BCTs of the beliefs included in a single message are identical.

There is a major caveat to all the above claims: they were proved under the assumption that there is no equality between beliefs, i.e., that the sum of costs for different values that can be assigned to a single variable are never identical, such that the best assignment at a particular stage is always clear. However, there is considerable theoretical and empirical evidence to indicate that when no steps are taken to ensure that this assumption holds, equal beliefs actually occur quite often. Thus, there is a clear motivation to enforce the condition of "no equality", in order for the results that were obtained under this assumption to stand. Two attempts to prevent belief equality have been published. The first is *value propagation*, commonly used within complete inference algorithms such as DPOP [12], but also used in versions of max-sum, e.g., Rogers et al. [14], Zivan et al. [22]. However, although this method is useful when the algorithm converges, it does not overcome pathologies that are triggered by belief equality during the algorithm's run, e.g., when solving graph coloring problems [22]. The second method for avoiding belief equality was presented by Farinelli et al. [4]. These authors proposed the addition of unary constraints with randomly selected costs, which are orders of magnitude smaller than the problem's constraint costs. These are intended to reduce the probability of belief equality to a negligible level. 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 to solve 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 on min-sum in particular, by means of the following contributions:

1. We establish conditions that lead to belief equality 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. [4] does not prevent ties even in a single-cycle graph.

Our results provide the first theoretical evidence that equalities cannot be avoided using the standard methods when applying min-sum to graphs that include cycles. This finding indicates that new theoretical analysis will be needed to tackle such settings. Our empirical results demonstrate that when the algorithm does not converge to the optimal solution, the rate of occurrence of assignment equality is high.

The rest of the paper is organized as follows: We first provide background information on graphical models and belief propagation, and we present the existing theoretical knowledge regarding the convergence properties of belief propagation. Next, we present preliminaries followed by our theoretical results. Then, we conduct an empirical evaluation in which we examine how often the algorithm fails to converge, and for these cases, we determine the rates of belief equality and assignment equality. We end with conclusions and suggestions for future work.

## 2. Background

In this section we present background information on the graphical models to which our results apply. In addition, we present the preliminaries of constraint optimization problems (COPs) and the version of belief propagation used to solve them – the min-sum algorithm. We also discuss the conditions for convergence on single-cycle graphs, as presented in Forney et al. [6].

### 2.1. Graphical models

Graphical models, such as Bayesian networks or constraint networks, are widely used representation frameworks for representing and solving optimization problems. The graph structure is used to capture dependencies between variables [10]. Our work extends the theory established by Weiss [19], who sought to determine the maximum a posteriori (MAP) assignment using the max-product version of belief propagation. The relationship between MAP assignment and constraint optimization is well established [10,4]. Thus, results that apply to the use of the max-product method to determine the MAP assignment also apply to the use of max/min-sum to

solve constraint optimization problems – and vice versa [15]. Without loss of generality, we focus here on constraint optimization, since it is more common in the AI literature. Our results apply to any version of a problem that can be represented by a graphical model and solved by belief propagation, as do the results of Weiss [19].

### 2.2. Constraint optimization

A constraint optimization problem, or $COP$, is a tuple $\langle \mathcal{X}, \mathcal{D}, \mathcal{R} \rangle$ in which $\mathcal{X}$ is a finite set of variables $\{X_1, X_2, \ldots, X_m\}$ and $\mathcal{D}$ 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), where each constraint $R_j \in \mathcal{R}$ defines a non-negative *cost* for every possible combination of values assigned to a set of variables, and it takes 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 every entry is a cost. Specifically, the entry $e_{x,y}$ is the cost incurred when $X_i$ is assigned the value $x$ and $X_j$ is assigned the value $y$. A *binary COP* is a COP in which all constraints are binary. The *cost of a partial assignment* (PA) is the sum of all constraints applicable to PA. A *complete assignment* (or a *solution*) is a partial assignment that includes all the variables of the COP. An *optimal solution* is a complete assignment with minimal cost. For simplicity, we concentrate solely on binary COPs.

### 2.3. Min-sum belief propagation

Min-sum operates on a *factor graph*, which is a bipartite graph in which the nodes represent variables and constraints [9]. Each variable-node, representing a variable of the original COP, is connected to all function-nodes that represent the constraints that involve it. Similarly, each function-node is connected to all variable-nodes that 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 computations. Min-sum can be applied to both distributed settings, where each node's role is performed by an autonomous agent (in such cases, the problem is referred to as a $DCOP$, i.e., a distributed COP), and to centralized settings, in which the activities of all nodes are managed by a single entity.

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

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

where $Q^i_{X \rightarrow F}$ is the message that variable-node $X$ intends to send to function-node $F$ at iteration $i$, $F_X$ is the set of function-node neighbors of variable-node $X$, and $R^{i-1}_{F' \rightarrow X}$ is the message that was sent to variable-node $X$ by function-node $F'$ at iteration $i-1$. The term $\alpha$ is a constant that is subtracted from each of the beliefs 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 $R^i_{F \rightarrow X}$ is sent from function-node $F$ to variable-node $X$ at iteration $i$ and includes a belief for each value $x \in D_X$. It is formulated as follows:

$$R^i_{F \rightarrow X} = \min_{PA_{-X}} cost(\langle X, x \rangle, PA_{-X}), \tag{2}$$

where $PA_{-X}$ is a possible combination of values assigned to the variables involved in $F$ excluding $X$. 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 given by:

$$f(a) + \sum_{X' \in X_F, X' \neq X, \langle X', x' \rangle \in a} (Q^{i-1}_{X' \rightarrow F})_{x'}, \tag{3}$$

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 $(Q^{i-1}_{X' \rightarrow F})_{x'}$ is the cost that was received in the message sent from variable-node $X'$ at iteration $i-1$, for the value $x'$ that is assigned to $X'$ in $a$. Variable-node $X$ selects its value $\hat{x} \in D_X$ following iteration $k$ as follows:

$$\hat{x} = \arg\min_{x \in D_X} \sum_{F \in F_X} (R^k_{F \rightarrow X})_x. \tag{4}$$

### 2.4. Single-cycle factor graphs

For a single-cycle factor graph, we know that if belief propagation converges, it converges to the optimal solution [6,19]. Moreover, when the algorithm does not converge, it periodically changes its set of assignments. To explain this behavior, Forney et al. [6] show

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

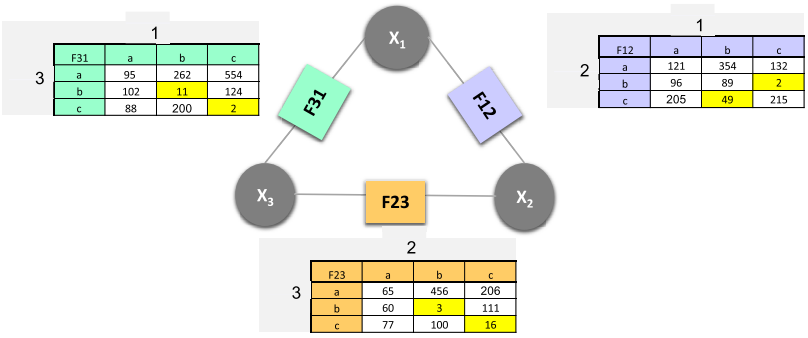



**Fig. 1.** A single cycle factor graph with three nodes, where each variable has a domain size of three. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

the similarity of the performance of the algorithm on a cycle to its performance on a chain-structured graph, whose nodes are similar to the nodes in the cycle, but whose length is equal to the number of iterations performed by the algorithm. Consider a sequence of messages starting at the first node of the chain and heading towards the other end. Each message conveys beliefs accumulated from the costs added by function-nodes. Specifically, each function-node adds a cost to each belief, which is the constraint cost of a pair of values assigned to its neighboring variable-nodes. Each such sequence of cost accumulations (i.e., each route) must become periodic at some point, and the minimal belief is generated by the minimal periodic route. If this periodic route is consistent, i.e., if the set of assignments implied by the costs within it contains the same value for each variable, the algorithm converges and the implied assignment is the optimal solution; otherwise, it does not converge [6]. Our results demonstrate that there are cases in which 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 from each variable's domain, and the beliefs for these values periodically become equal. This phenomenon cannot be avoided using the unary constraint method proposed in Farinelli et al. [4].

**Example 1.** Fig. 1 presents a single-cycle factor graph with three variable-nodes $X_1$, $X_2$ and $X_3$ and three function-nodes $F_{12}$, $F_{23}$ and $F_{31}$. Each variable has three values in its domain ($a$, $b$ and $c$) and thus the dimensions of the cost tables held by the function-nodes are $3 \times 3$. At every synchronous iteration, each of the nodes sends two messages, one to each of its neighbors. Each message consists of a vector of beliefs (costs) of size 3, one for each of the values of the variable involved (either sending or receiving). At the first iteration, all messages sent by all variable-nodes contain zero costs. The messages sent by the function-nodes represent, for each possible value of each of their neighboring variable-nodes, the minimal cost they can offer for that value, according to the cost table. For example, at the first iteration, function-node $F_{12}$ sends the message $\langle 121, 2, 49 \rangle$ to variable-node $X_2$, since these are the minimal entries in each of the rows of its cost table. At the second iteration, variable-node $X_2$ sends the same message to function-node $F_{23}$. At the third iteration, $F_{23}$ sends the message $\langle 186, 5, 65 \rangle$ to $X_3$, which is derived as follows: for every row in the cost table, the beliefs in the message received from $X_2$ at the previous iteration are added, e.g., 121 is added to 65, 2 is added to 456 and 49 is added to 206. Thus, 186, which is the minimal of the three, is the first belief to be included in the message that will be sent to $X_3$. The beliefs 5 and 65 are selected similarly. At the same iteration, $X_3$ receives from its other neighbor, $F_{31}$, the message $\langle 191, 60, 4 \rangle$. If $X_3$ needs to select an assignment at this iteration, it sums the two vectors that it has received, resulting in $\langle 377, 65, 69 \rangle$, and thus it selects the value $b$ since this results in the minimal cost.

**Example 2.** Fig. 2 presents the same factor graph as that presented in Fig. 1 with the addition of unary constraints (as suggested in [4]). Each vector of unary constraints adjacent to a variable-node specifies a very small additional cost for each of its value assignments, $a$, $b$ and $c$ respectively.

### 2.5. Convergence of min-sum

The convergence of min-sum has been the focus of a number of studies in the graphical models and multi-agent systems communities. With the exception of the studies on tree-structured graphs and single-cycle graphs, in which the conditions for convergence have been well established [11,19,6], most of the attempts to identify the conditions for convergence of the algorithm on general graphs, with an arbitrary number of cycles, have suggested applying some revisions to the original algorithm in order to trigger convergence.

Ihler et al. [8] analyze belief propagation (BP) from the unique point of view that each message sent at each iteration conveys a noisy version of the true belief. Thus, analyzing the error trends can shed light on the convergence of the algorithm. A number of studies have investigated how the convergence of BP is related to physical properties and system dynamics. Tatikonda and Jordan [17] study the relationship between the convergence of BP and Gibbs measures. Similar to the approach suggested in [20,23], they make use of the unwrapped tree. However, they use Gibbs measures to establish conditions for convergence. Heskes [7] studies the convergence of BP by identifying its similarity to a Bethe free energy function, a tractable approximation to the exact variational Gibbs-Helmholtz free energy equation. The approximation method simulates a function of beliefs with certain properties that guarantee convergence. None of the aforementioned studies addresses the possibility of belief or assignment equality.

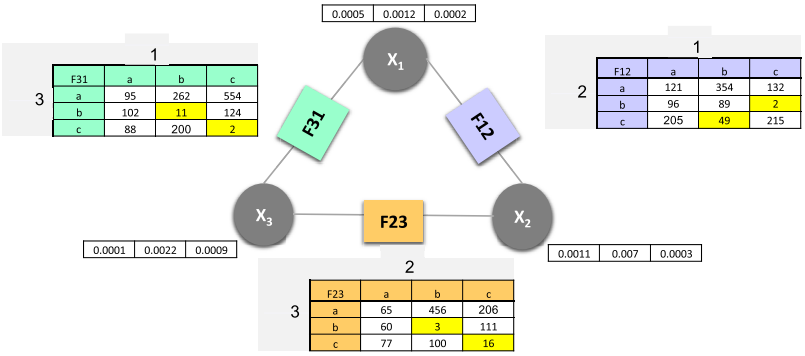



**Fig. 2.** A single cycle factor graph with three nodes, where each variable has a domain size of three, with unary constraints.

In the DCOP literature, there have been a number of attempts to revise the factor-graph structure in order to guarantee convergence. One approach involves removing edges from the factor graph until it is reduced to a tree-structured graph [14]. Here, the authors use the unary constraints tie breaking method that they proposed in [4], which is indeed effective when convergence is guaranteed. The resulting algorithm, referred to as bounded max-sum, enables calculation of a bound on the distance between the quality of the reported solution and the cost of the optimal solution. In [22], the authors attempt to guarantee convergence without removing edges from the factor graph by transforming the factor graph into a directed acyclic graph. In common with other studies, the study invokes the unary constraints method for preventing ties.

## 3. Preliminaries

Our analysis will provide 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 at time $t > 0$ are selected according to the messages R that are sent to the variable-nodes from their function-node neighbors at time $t-1$.

For every pair of constrained variables, $X_i$ and $X_j$, for each $x \in D_i$, $x' \in D_j$, we 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 entry in the cost table held by the function-node representing the constraint between $X_i$ and $X_j$.

Forney et al. [6] 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. [6] further proved that if the algorithm converges, i.e., if $v = 1$, then the single repeated state is the optimal solution. Intuitively, the optimality of the solution follows from the optimality of belief propagation on trees in general and on chains in particular [11]. If we were to select an arbitrarily long chain representing the solution process of belief propagation on the single-cycle graph, then it would include the assignment that is derived from the repeated set periodically for the number of iterations that we chose. If this assignment were not optimal, it 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.** (proposed in [23]) A backtrack cost tree (BCT) for a belief $b^k$ (i.e., BCT($b^k$)) is a hierarchical structure that specifies the constraint costs that were summed in order to generate $b^k$ at iteration $k$. A belief is generated by summing costs (beliefs) that are sent in messages R. Thus, if $b^k$ is a belief sent in message R at iteration $k$, then for $k > 1$, it is the sum of the beliefs that were sent in the messages at iteration $k-2$, together with an additional cost $C_{(X_i=x,X_j=x')}$, which is an entry in the cost table of the function-node that sends the message R that includes $b^k$. Similarly and recursively, each belief $b^{k-2}$ that was sent in a message at iteration $k-2$ and was used to generate $b^k$, was generated by summing the beliefs sent at iteration $k-4$ (assuming $k > 3$) together with an entry in the cost table of the function-node that sends the message that includes $b^{k-2}$. 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 (see example in Fig. 3, showing a BCT for the cost sent from $F_{31}$ to $X_1$ at iteration 12 for value assignment $b$, when solving the problem presented in Fig. 1). The BCT starts by specifying the cost sent from function-node $F_{12}$ to variable-node $X_2$ at the second iteration. It proceeds by specifying the accumulated cost from previous iterations plus the entry that is added at the current iteration. It is apparent that in a chain-structured BCT representing the process on a single-cycle factor graph, only the function-nodes add to the costs of the BCT, while variable-nodes simply pass on the costs they receive. The following definition allows us to focus on the cost table entries that are summed in a BCT.

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

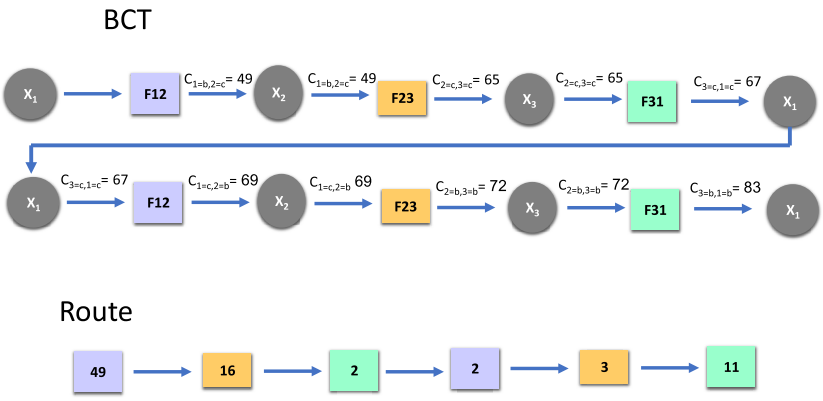



**Fig. 3.** BCT and route of the belief sent to variable-node $X_1$ for value $b$, according to the graph in Fig. 1.

The bottom of Fig. 3 depicts the route that corresponds to the BCT presented at the top of the figure. It shows the series of costs that are summed in order to generate the belief sent from $F_{31}$ to $X_1$ at the 12'th iteration for value assignment $b$. We say that the route *visits* a cost table entry when this entry is added to the route. We will denote the set of entries that were added to a route $r$ (i.e., the entries that the route $r$ is composed of) by $E_r$. For simplicity, we will say that an entry $e$ is included in $r$ when $e \in E_r$.

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

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 $k$'th and $(k+1)$'th 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. [6], we know that there is a time $t_0$ such that for any time $t > t_0$, the minimal route corresponds to the minimal normalized sequence of entries (an interval) that repeats itself periodically. When the size of the periodic interval is equal to the number of function-nodes in the graph, this means that the algorithm has converged and the minimal assignment is the optimal solution (since the assignment of every variable is always the same, whenever the route "visits" it).

We will refer to the set of cost table entries $C_{(X_i=x,X_j=x')}$ that are included in the minimal route after $t_0$ as the set $M$, and the assignment implied by a single interval of the periodic route that follows $t_0$ and visits each of the members in $M$ as $A^M$. In the factor graph presented in Fig. 1, the minimal route (as depicted at the bottom of Fig. 3) includes all the entries in the cost table that are highlighted in yellow. Consider a route on the factor graph depicted in Fig. 2 that includes all members in $M$ (highlighted in yellow). Each member $m \in M$ is an entry in a cost table of some function-node $F$, and there is a value assignment implied by $m$ for the variable-node that follows $F$ according to the route's order. For this value assignment, there is a corresponding unary constraint that is added to the accumulated cost. For example, for a clockwise route, if the entry $\langle b, b \rangle$ of the cost table of $F_{23}$ is added to the route, this implies that the cost 0.0022 for $X_3 = b$ is also added to the accumulated cost. Thus, every time the route includes this entry in $M$, the cost of the route is incremented by 3.0022. In other words, if two routes include the same cost table entries, then the unary costs added to them are the same as well.

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 a message $R$ in Formulas (2) and (3). The process is demonstrated in Example 1.

The implication of this observation is that the number of possible routes of length $m$, assuming that each variable has a domain size $d$, is $2nd^m$ and not $2nd^{2m}$, since there are $n$ different starting points and two possible directions in which to proceed. Then, for every function-node, there are $d$ options to choose from.

## 4. Conditions for belief and value assignment equality

We first define belief equality and value assignment equality, and then establish the conditions that trigger them.

**Definition 4.** We use the term **belief equality** to refer to the case where a message includes two or more identical beliefs (costs).

**Definition 5.** We use the term **value assignment equality** (or **assignment equality**) to refer to the case where, at some iteration, the sum of beliefs received by a variable-node is identical for two or more of its values.

To illustrate the difference between the two, consider a variable-node $X_i$ representing a variable with two values in its domain, $a$ and $b$. At some iteration $t$, $X_i$ receives two messages from its neighboring function-nodes, $\langle 0,2\rangle$ and $\langle 0,0\rangle$. The first of these does not involve belief equality, but the second does. The sum of the two messages is 0 for $a$ and 2 for $b$; thus, variable $i$ does not exhibit value assignment equality at this iteration. Now, consider another iteration $t' \neq t$ in which $X_i$ receives the messages $\langle 0,2\rangle$ and $\langle 2,0\rangle$. In this case, neither of the messages exhibits belief equality, but they do result in value assignment equality since the sum of the messages produces a value of 2 for $a$ and 2 for $b$, and thus, the variable is indifferent regarding the two possible assignments. It is easy to see that if all messages received by a variable-node at a single iteration entail belief equality for the same set of values, then there must be assignment equality for these values at that iteration as well. For example, if $X_i$ receives messages $\langle 0,0\rangle$ and $\langle 0,0\rangle$ at iteration $t$, then obviously, both belief equality and value assignment equality exist. Note that when there are more than two values in a domain, belief equality in each of the messages received at an iteration does not necessarily imply that there is value assignment equality, e.g., if $D_i$ includes values $a$, $b$ and $c$, and at some iteration $t$, the messages received are $\langle 0,0,2\rangle$ and $\langle 0,3,3\rangle$, then there is belief equality in both messages, but there is no assignment equality, since the sum of the messages is 0 for $a$, 3 for $b$ and 5 for $c$.

In order to study the cause of the belief equality phenomenon (i.e., messages that include identical costs for at least two value assignments), we analyze the components from which these beliefs were constructed. That is, we determine which entries in the constraint cost tables are summed in the process of generating equal beliefs. Since we can use the unary constraint method proposed in Farinelli et al. [4], 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 that are summed are not identical – there is a difference between beliefs, and therefore, there is a minimal belief in each message.

**Lemma 1.** *When the algorithm does not converge, there must be at least one function-node $F$ with two cost table entries in $M$ (the set of cost table entries in the minimal route, which the algorithm visits at every iteration $t > t_0$).*

**Proof.** When it is not the case that there is at least one function-node with two cost table entries in $M$, then, in the minimal assignment, there is a single value assignment for each variable, and this means that the algorithm converges. □

**Proposition 1.** *When the algorithm does not converge, and assuming there are no tied beliefs, then any two entries in $M$ that 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 $F_{ij}$ representing the constraint between $X_i$ and $X_j$, consider the message that is sent from $F_{ij}$ to $X_j$. In this message, there is a minimal belief corresponding to the assignment $X_i = x$. Thus, the relevant entries to extend this route (i.e., the route that implies that $X_i = x$) are the entries in the vector $C_{(X_i=x,X_j=\cdot)}$, of which only one is minimal. Thus, if more than one of the entries in $C_{(X_i=x,X_j=\cdot)}$ are included in $M$, this contradicts the minimal route property.

Since the algorithm operates in both directions, a similar argument rules out the possibility that there are two entries in $M$ whose implied assignments would result in the same value being assigned to $X_j$. □

**Proposition 2.** *In the minimal periodic route that begins after $t_0$, in every periodic interval, each cost table entry that is included in $M$ is visited exactly once, i.e., it will be visited again only after all other members of $M$ have been visited exactly once.*

**Proof.** Assume that the proposition does not hold, and thus, it is possible that an entry in $M$ will be visited twice before another entry in $M$ is visited. Then, there must be an entry $e$ in some function-node cost table (e.g., $C_{X_i=x,X_j=y}$), where entry $e$ is included in $M$, such that the route includes more than one entry that directly follows it. In other words, there must be at least two entries, $e'$ and $e''$, such that the entry that immediately follows $e$ in the minimal route will alternate between $e'$ and $e''$ (e.g., $e' = C_{X_j=j,X_k=z}$; $e'' = C_{X_j=j,X_k=z'}$). However, this would mean that $e'$ and $e''$ are both included in $M$ despite having the same value assignment for one of their variables, and this would contradict Proposition 1. □

A corollary that immediately follows from Proposition 2 is that the number of entries in each function-node cost table that are visited in a periodic interval of the minimal route is equal to the number of values that belong to the same domain and are assigned to the same variable in $A^M$. We denote this number by $\bar{d}$. We note that $\upsilon > 1$ if and only if $\bar{d} > 1$, but it is not always the case that $\upsilon = \bar{d}$. Moreover, the size of a periodic interval after $t_0$ is $2\bar{d}n$, where $n$ is the number of variable-nodes in the single-cycle graph.

**Proposition 3.** *Belief equality exists if and only if there are two beliefs in a single message, $b$ and $b'$, with corresponding routes $r_b$ and $r_{b'}$, such that $E_{r_b} = E_{r_{b'}}$.*

[3] As Fig. 1 shows, yellow cells (which are the members of $M$) included in the cost table of the same function-node do not occupy the same row or column.

**Proof.** Assume that there exist two beliefs $b$ and $b'$ that are included in the same message, with corresponding routes $r_b$ and $r_{b'}$, such that $E_{r_b} = E_{r_{b'}}$. Obviously, the sum of entries in these two routes is identical.

Assume that there is belief equality. Thus, there are two equal beliefs that are included in the same message. Since we use the unary constraints method of Farinelli et al. [4], it must hold that the two routes are composed of the same cost table entries, in order to generate equal sums of their components. □

**Proposition 4.** *Value assignment equality exists if and only if, at some iteration of the algorithm $t > 0$, there are two values in the same domain, $x$ and $x'$, $x \neq x'$, with corresponding routes $r^{cw}_x$, $r^{ccw}_x$, $r^{cw}_{x'}$ and $r^{ccw}_{x'}$, such that $E_{r^{cw}_x} \cup E_{r^{ccw}_x} = E_{r^{cw}_{x'}} \cup E_{r^{ccw}_{x'}}$.*

**Proof.** The arguments are similar to those stated in the proof of Proposition 3, with the costs for assigning $x$ and $x'$ composed of the clockwise and the counter-clockwise beliefs. □

### 4.1. Immediate convergence

In the next part of our discussion, we assume that $t_0 = 0$, i.e., that the algorithm immediately begins to traverse a minimal route. Note that we do not assume that the algorithm converges, i.e., that $\upsilon = 1$, but rather that the minimal periodic route of size $2\bar{d}n$ for each period begins right away.

**Theorem 1.** *When the algorithm runs on a single-cycle graph and does not converge (i.e., $\bar{d} > 1$), and when $t_0 = 0$, for any natural number $k$, after $2k\bar{d}n$ iterations, all beliefs sent, which correspond to the values that are assigned in $A^M$, are equal.*

**Proof.** We know from Proposition 2 that after $t_0$, every $2\bar{d}n$ iterations, the minimal route visits each of the entries in $M$ exactly once. Since $\bar{d} > 1$, we know that for each function-node, multiple members of $M$ are visited that do not occupy the same line or row, i.e., they imply distinct value assignments. Thus, if $t_0 = 0$, then over the course of the first $2\bar{d}n$ iterations, each of the entries in $M$ will be visited exactly once by BCTs that follow the minimal route; this is then repeated every $2\bar{d}n$ iterations. This means that at the end of every such period, the beliefs corresponding to value assignments in $A^M$ are composed of the same elements (function-node entries), and thus, according to Proposition 3, they are equal. □

An immediate corollary arising from Theorem 1 is that when $\bar{d} > 1$ and $t_0 = 0$, for any natural number $k$, after $2k\bar{d}n$ iterations, each variable-node will exhibit assignment equality between $\bar{d}$ of the values in its domain.

A second corollary is that for $t_0 > 0$, if for some variable $X_i$ at $t_0$, the difference between beliefs $b^{t_0}_x$ and $b^{t_0}_{x'}$ for two values $x \in D_i$ and $x' \in D_i$ is $\delta$ (e.g., $b^{t_0}_x$ - $b^{t_0}_{x'} = \delta$), then at any iteration $t$ such that $t - t_0 = 2k\bar{d}n$, the difference between the beliefs for the two values will be $\delta$.

**Example 3.** In the following example, we revisit the rather simple factor graph presented in Fig. 1, where the graph has three variable-nodes, each with a domain size of 3, i.e., $n = 3$ and $d = 3$.

The entries highlighted in yellow in the cost tables presented in Fig. 1 are included in the set $M$, i.e., they are visited repeatedly in the minimal route, after 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 proved in Theorem 1, the costs of all beliefs corresponding to the values in the minimal assignment, where these values are the coordinates of all the entries in $M$, will be equal after visiting each of the members of $M$ once. Since there are $2n$ nodes in the single-cycle factor graph, the number of iterations required to visit the six members of $M$ is 12.

Fig. 3 presents the BCT and corresponding route for the belief sent at the 12'th iteration to variable-node $X_1$ for its value $b$. The BCT accumulates the members of $M$ in the order that is displayed in the route. During the next 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 also be visited by the BCT and route of the belief sent to variable-node $X_1$ for its value $c$. Thus, belief equality will arise every 12 iterations. Fig. 4 presents the four routes, two clockwise and two counterclockwise, that result in belief equality in each direction and in assignment equality between values $b$ and $c$ for variable-node $X_1$.

Fig. 5 presents the routes for the graph that includes unary constraints, shown in Fig. 2. Each node in the route is obtained by summing the selected entry in the function-node cost table and the corresponding unary constraint for the value of the variable-node that follows it.

For the factor graph in Fig. 1, if we replace the entry in $F_{12}$ for $X_1 = b, X_2 = b$ (currently 89) with a number smaller than 27.5, the solution $X_1 = b, X_2 = b, X_3 = b$ is optimal, and the set $M$ includes the entry $\langle b, b\rangle$ for each of the function-nodes. In this case, the minimal route is unique and there is no belief equality. Fig. 6 presents the BCT and minimal route to which min-sum converges after changing the entry $\langle b, b\rangle$ in $F_{12}$ from 89 to 23. Note that the accumulated cost of the route in this case is 74, which is smaller than 83.

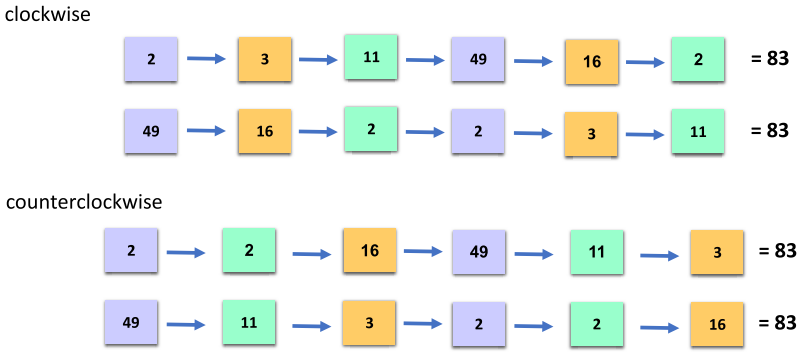



**Fig. 4.** Four routes for variable-node $X_1$ in the factor graph presented in Fig. 1, corresponding to values $b$ and $c$, which result in both belief equality and assignment equality after 12 iterations.

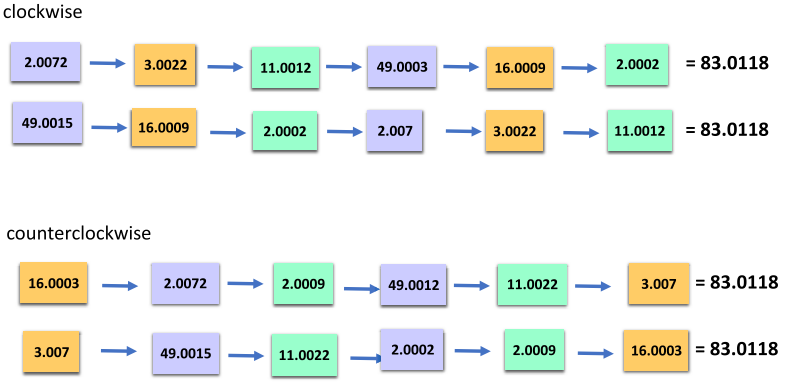



**Fig. 5.** Four routes for variable-node $X_1$, for values $b$ and $c$, that result in belief and assignment equality after 12 iterations, according to the graph in Fig. 2.

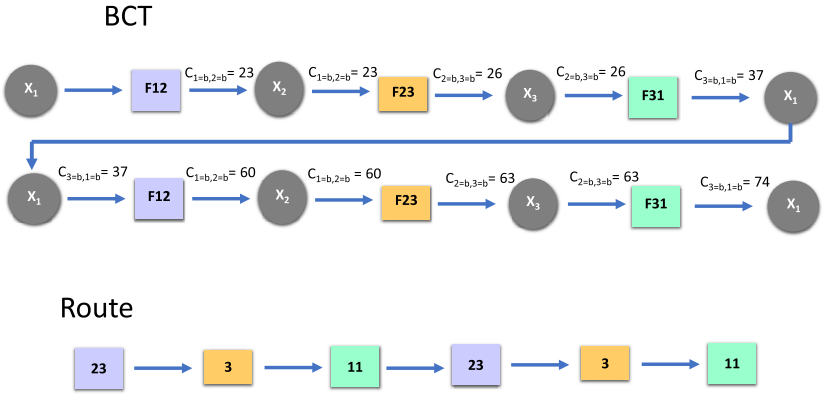



**Fig. 6.** BCT and route of the belief sent to variable-node $X_1$ for value $b$, after changing the graph in Fig. 1 so that the entry $\langle b, b\rangle$ of $F_{12}$ is equal to 23.

### 4.2. Non-immediate convergence

We now turn to discuss the case where $t_0 > 0$, i.e., the algorithm does not follow a periodic path straight away, but rather, it does so after several iterations [19,6].

**Definition 6.** For each route $r$, we refer to the part of the route that comes before $t_0$ as the *tail* and we denote it by $tail_r$.

**Example 4.** If we replace in Fig. 1 the entry $\langle b, b\rangle$ in the cost table of function-node $F_{12}$ with the value 42, then we have a "tail", since in the first message sent from $F_{12}$ to variable-node $X_1$, the entry $X_2 = b, X_1 = b$ will be selected, although it is not a member of $M$. This tail prevents belief equality in the counterclockwise direction (but not in the clockwise direction). As a result, there is no assignment equality, as can be seen from Fig. 7, which presents the relevant routes.

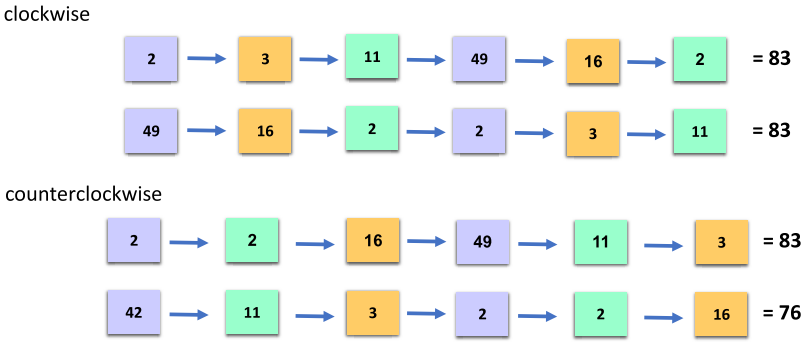



**Fig. 7.** Four routes for variable-node $X_1$, for values $b$ and $c$, according to the graph in Fig. 1 after replacing the cost in $F_{12}\langle b, b\rangle$ with 42.

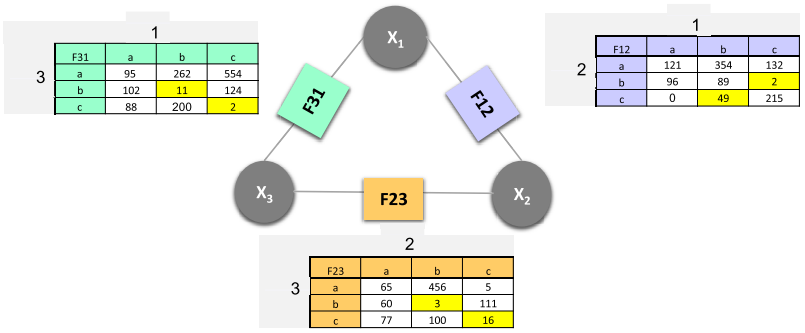



**Fig. 8.** A different (second) single cycle factor graph with three nodes, where each variable has a domain size of three.

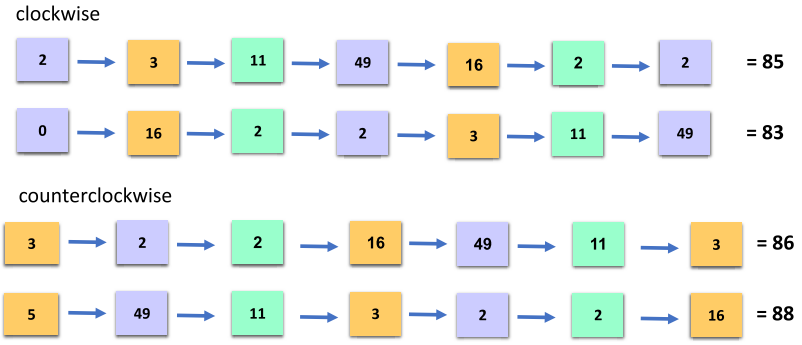



**Fig. 9.** Four routes for variable-node $X_1$, for values $b$ and $c$, on the graph presented in Fig. 8. The routes do not result in assignment equality since they are not composed of the same entries.

The cycle presented in Fig. 8 includes, in the cost tables of $F_{12}$ and $F_{23}$, entries that are not in $M$, although their costs are minimal either in a row or in a column. The result is that there are tails of length 1 in each direction, which prevent belief equality. Moreover, as demonstrated in Fig. 9, although after 14 iterations (routes of length 7), the sum of beliefs for each value is 171, there is no assignment equality, because these routes are not constructed from the same entries.

**Example 5.** In the final example, we present the periodic routes in Fig. 11, starting at $t_0$ (which is iteration 12), when solving the graph presented in Fig. 10. The tails of these routes, which are presented in Fig. 12, prevent belief equality, but do not prevent assignment equality. This is because the set of entries in the route tails that contribute to the calculation of the beliefs for value $a$ (i.e., the union of the top clockwise and top counter-clockwise route tails) is identical to the set of entries in the two route tails (clockwise and counter-clockwise) used to calculate the beliefs for value $c$ (the bottom route tails).

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 depends on whether or not these tails are composed of the same components.

**Theorem 2.** *When min-sum is applied to a single-cycle graph, belief equality will occur if the following conditions hold:*

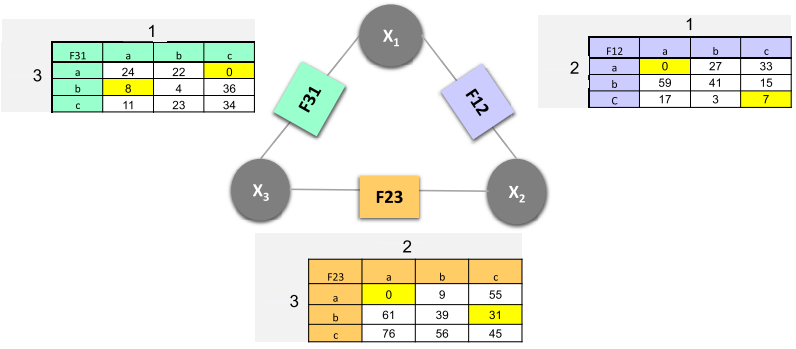



**Fig. 10.** A different (third) single cycle factor graph with three nodes, where each variable has a domain size of three.

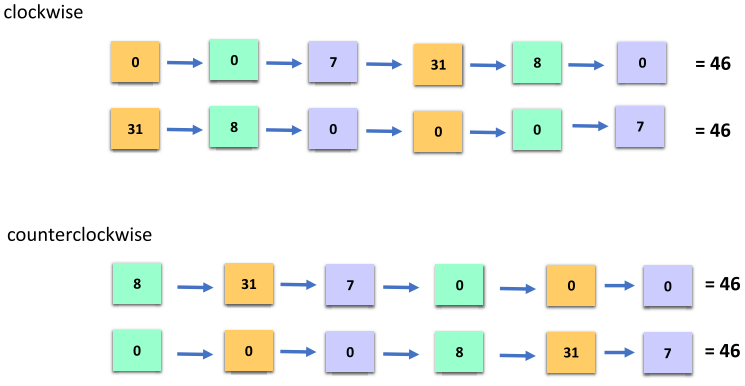



**Fig. 11.** Four routes for variable-node $X_1$, for values $a$ and $c$, on the graph presented in Fig. 10, starting at $t_0$, all with equal costs.

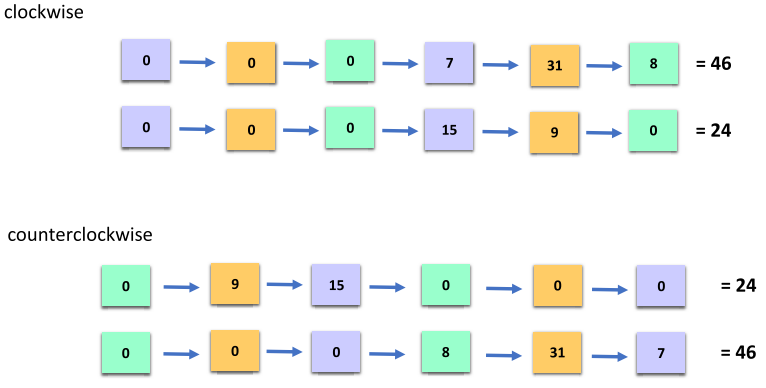



**Fig. 12.** Four tail routes that precede the routes presented in Fig. 11, on the graph presented in Fig. 10. The tail routes prevent belief equality but not assignment equality.

1. $\bar{d} > 1$.
2. *The algorithm performs $t$ iterations such that $t - t_0 = 2k\bar{d}n$ for any $k \in \mathbb{N}$, i.e., the number of iterations since convergence to a periodic pattern is $2k\bar{d}n$.*
3. *There is a variable $X_i$ such that for some two values $x \in D_i$, $x' \in D_i$ $(x \neq x')$, both of which are assigned to $X_i$ in $A^M$ (note that this is possible since $\bar{d} > 1$), the variable-node that represents $X_i$ receives beliefs through routes $r_x$ and $r_{x'}$, both of which have tails ($tail_{r_x}$ and $tail_{r_{x'}}$, respectively). Moreover, $E_{tail_{r_x}} = E_{tail_{r_{x'}}}$.*

**Proof.** Assume the three conditions hold. Then, according to Theorem 1 and its second corollary, the routes of the beliefs corresponding to the value assignments in the minimal assignment after $t_0$ are equal (composed of the same entries). Thus, if their tails are also equal (composed of the same entries), then according to Proposition 3, the theorem holds. □

The following theorem establishes conditions for assignment equality.

**Theorem 3.** *When min-sum is applied to a single-cycle graph for $t$ iterations, and $t - t_0 = 2k\bar{d}n$ for any $k \in \mathbb{N}$ (i.e., the number of iterations since convergence to a periodic pattern is $2k\bar{d}n$), then assignment equality occurs if and only if the following two conditions hold:*

1. $\bar{d} > 1$.
2. *There is a variable $X_i$ such that for some two values $x \in D_i$, $x' \in D_i$ $(x \neq x')$, both of which are assigned to $X_i$ in $A^M$, the routes corresponding to these values are $r_x^{cw}$, $r_x^{ccw}$, $r_{x'}^{cw}$ and $r_{x'}^{ccw}$, and the following equation holds:*

$$E_{tail_{r_x^{cw}}} \cup E_{tail_{r_x^{ccw}}} = E_{tail_{r_{x'}^{ccw}}} \cup E_{tail_{r_{x'}^{cw}}}.$$

**Proof.** Assume the two conditions hold. Then, according to Theorem 1, the routes of all beliefs corresponding to the value assignments in the minimal assignment after $t_0$ are composed of the same components and sum up to identical costs every $2k\bar{d}n$ iterations. Thus, according to Proposition 4, the second condition implies assignment equality.

Assume that the unions of the sets of entries corresponding to the routes for two values in the same domain, both of which belong to the minimal assignment $A^M$, are identical. This implies that $\bar{d} > 1$; otherwise, there would be a single value from each domain assigned in $A^M$. Without loss of generality, assume that we are referring to values $x$ and $x'$ in the domain of variable $X_i$, and that both values are assigned to $X_i$ in $A^M$. Following Theorem 1 and its second corollary, we know that after $t_0$, every $2k\bar{d}n$ iterations, the sum of beliefs for each of the values $x$ and $x'$ will be identical, i.e.,

$$E_{r_x^{cw}} \setminus E_{tail_{r_x^{cw}}} = E_{r_x^{ccw}} \setminus E_{tail_{r_x^{ccw}}} = E_{r_{x'}^{cw}} \setminus E_{tail_{r_{x'}^{cw}}} = E_{r_{x'}^{ccw}} \setminus E_{tail_{r_{x'}^{ccw}}}$$

(According to Theorem 1, regardless of the starting point and direction, after $t_0$, every $2k\bar{d}n$ iterations, the set of entries that is added to each route is equal to $M$.)

Then, following the assumption and Proposition 4, it must hold that:

$$E_{tail_{r_x^{cw}}} + E_{tail_{r_x^{ccw}}} = E_{tail_{r_{x'}^{cw}}} + E_{tail_{r_{x'}^{ccw}}} \quad \square$$

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

## 5. Experimental evaluation

Having derived theoretical proofs, in this section, our aim is to gain a perspective on how frequently belief equality and assignment equality arise when min-sum is applied to a single-cycle factor graph. To do so, we created simulations in which we varied the size of the cycle, the size of the domain, and the structure of the constraint functions. We implemented the method proposed by Farinelli et al. [4] for avoiding equalities, by assigning random unary constraints, selected uniformly from the range $[10^{-8}, 10^{-4}]$. Since our claims deal with cases that do not converge, for each combination of properties used to generate simulation instances (e.g., domain size and cycle size), we generated random instances and solved them using min-sum until we had created 100 instances on which min-sum did not converge.

The single-cycle constraint graphs that we used in our simulation included four types of distribution from which the constraint costs were sampled. Specifically, for each pair of values assigned to every pair of neighboring variables, the cost was selected according to one of the following four options:

**Uniform distribution** Constraint costs were selected uniformly between 0 and 100.

**Poisson distribution** Constraint costs were selected according to a Poisson distribution with a mean of 50 (i.e., cost $\sim Pois(50)$).

**Index-based Poisson distribution** Constraint costs were selected using a Poisson distribution with a mean parameter that was determined while taking into consideration the variables' indexes; i.e., the costs selected for $X_i$ and $X_j$ were taken from $Pois(i * j)$, where $i, j \in \{1, 2, ..., n\}$, $n = |\mathcal{X}|$ and $i \neq j$.

**Preferred values** In this distribution, each agent employs a randomized process to select a set of preferred values, denoted as the set $PV_i$. The constraint cost associated with variables $X_i$ and $X_j$ reflects the level of satisfaction with the chosen values. Our assumption is that if the preferred values cannot be selected, the values that are adjacent to them are preferred next, and these, in turn, will incur smaller costs than values that are even further away. This type of cost distribution corresponds to realistic scenarios such as meeting scheduling. To be specific, we defined the preference cost as $PC_i(v_i, PV_i)$, which quantifies the level of satisfaction from the assignment $v_i$. $PC$ was calculated as follows: $\min_{k \in \mathrm{PV}_i} |k - \mathrm{v}_i|$. Then, the constraint cost corresponding to the entry $R_{ij}$ was sampled from a normal distribution where the standard deviation was set to 10 and the mean was calculated using the following formula: $mu_{ij} = 10 \cdot avg(PC_i(v_i, PV_i), PC_j(v_j, PV_j))$. We generated problems with a single preferred value and with three preferred values in each domain.

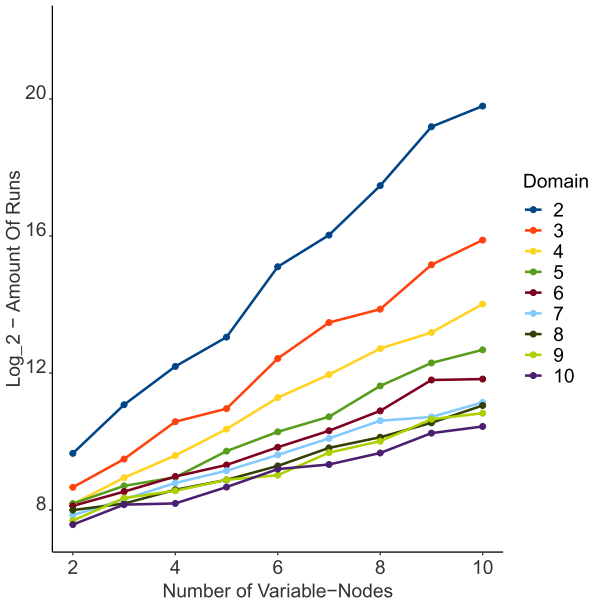



**Fig. 13.** Logarithmic scale of the number of instances solved before 100 instances on which min-sum did not converge were encountered, for single-cycle graphs in which constraints were sampled from a uniform distribution.

While we were limited in the selection of the graph structure (since we focus on single-cycle graphs in this research), the choice of different constraint structures serves to explore constraints with both abstract and realistic cost distributions. Poisson cost distributions simulate realistic scenarios in which preferences regarding the value of a variable are not uniformly distributed, but rather in close proximity to some reasonable mean. The index-based Poisson distribution simulates scenarios in which these preferences are subjective. The final option (preferred values) reflects realistic scenarios in which specific values are preferred over others, e.g., when scheduling meetings, participants commonly have preferred time slots, such as the morning or evening.

Fig. 13 presents (on a logarithmic scale) the number of instances solved, for each combination of cycle size (number of variable-nodes) and domain size, before 100 instances that did not converge were generated under uniformly sampled costs. The trends are clear: the larger the size of the single-cycle graph, the smaller the proportion of instances on which the algorithm does not converge. In contrast, the larger the domain size, the greater the proportion of instances that do not converge. While the number of instances required to find one hundred instances that do not converge increases with the size of the graph for all domain sizes, the growth is fastest for the smallest domain sizes.

Similar results were obtained for the other constraint-cost selection distributions (Fig. 14). For all of them, the differences between graphs are very small when the variables have large domains, but they become more prominent when variables have small domains (the greatest difference being observed when comparing problems with domains sizes of 2 and 3).

The results for the problems in which the constraints were chosen using the preferred values approach were more consistent (see Fig. 15). It seems that when some values have lower costs (on average) than others, the size of the domain has a smaller effect on the number of instances on which min-sum does not converge.

Fig. 16 shows the number of instances in which at least one message with belief equality was generated, among the 100 single-cycle factor graphs with uniformly-selected constraint costs on which min-sum did not converge. It is apparent that regardless of the size of the cycle and the size of the domain, on average, the algorithm exhibits belief equality in approximately half of the instances. The lowest rate of occurrence of belief equality is approximately 25%, and is observed for the minimal domain size (2) and the minimal cycle size (2). The highest rate is close to 75%, and is observed for cycles with 2 variable-nodes and a domain size of 5. While there is no consistent trend, the smallest number of instances in which belief equality was generated is found for problems with the smallest domain size. In addition, for all domain sizes larger than 2, the highest rate of belief equality is found for problems with 2 variable-nodes in the cycle, while the lowest rate is usually observed for problems with more than 8 variable-nodes in the cycle.

Fig. 17 presents the number of instances in which assignment equality was generated among the same 100 instances. It is clear that the frequency of occurrence of assignment equality is considerably higher than that of belief equality. While the highest number of instances in which belief equality occurred is 75, the highest number of instances in which assignment equality occurred is close to 100. Another clear difference is the effect of the cycle size. In the case of assignment quality, for all problems with domain sizes larger than 3, a larger cycle size generally results in a larger number of instances in which assignment equality was generated, while the opposite is observed for belief equality (i.e., a lower rate of belief equality with increasing cycle size). A third difference relates to the trends with respect to domain size and cycle size; these are more consistent for assignment equality than for belief equality. Finally, in the case of assignment equality, the rate of occurrence is high for small domain sizes, irrespective of the cycle size, while for large domain sizes, assignment equality is more frequent at larger cycle sizes. In contrast, domain size does not have a substantial effect on the rate of occurrence of belief equality.

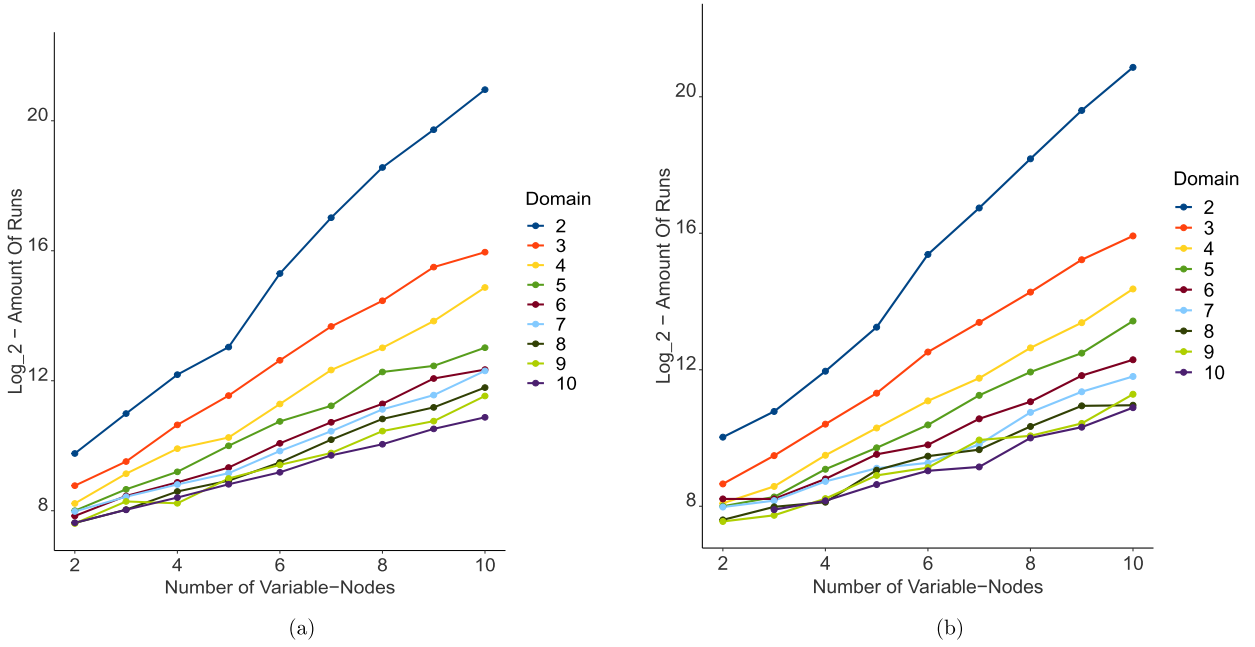



**Fig. 14.** Logarithmic scale of the number of instances solved before 100 instances on which min-sum did not converge were generated, for single-cycle graphs in which constraint costs were sampled from (a) a Poisson distribution and (b) an index-based Poisson distribution.

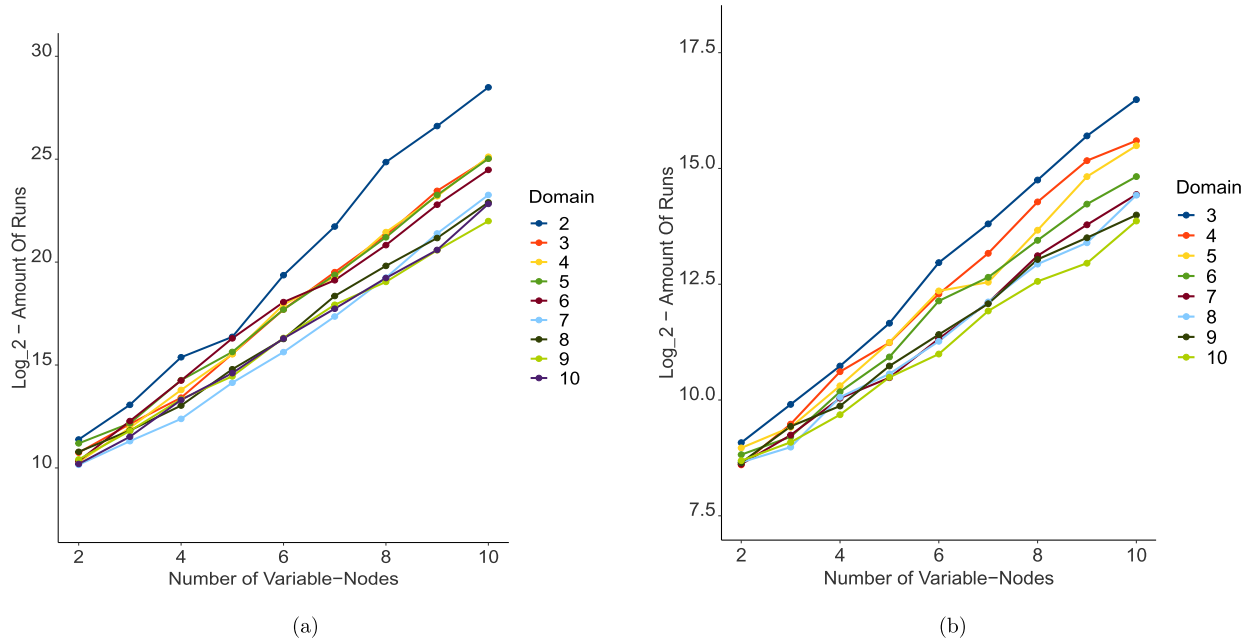



**Fig. 15.** Logarithmic scale of the number of instances solved before 100 instances on which min-sum did not converge were generated, for problems in which the constraints were generated via the preferred values method with (a) a single and (b) three preferred values in each domain.

The difference between the frequency of occurrence of belief equality and of assignment equality seems to be a consequence of the difference between Theorems 2 and 3. The main difference between these theorems relates to the conditions of equality of the tails (i.e., the routes before $t_0$). While for belief equality, the condition requires equality of tails for two values in a variable's domain, for assignment equality, it requires that the unions of the two tails (clockwise and counterclockwise) are the same. The simulations reveal that the latter condition appears to be much easier to achieve, demonstrating that belief equality is not a precondition for the existence of assignment equality. Similar results were obtained for the remaining constraint-cost structures (see Appendix A).

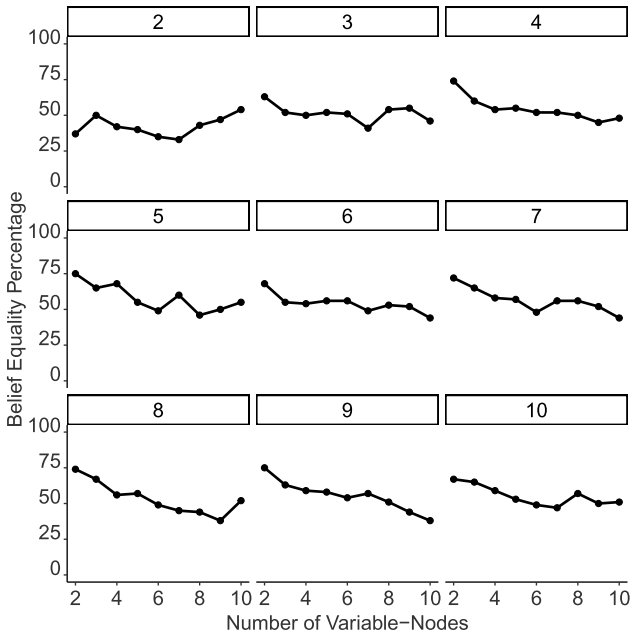



**Fig. 16.** Number of instances among the 100 that did not converge in which belief equality was generated as a function of the number of variable-nodes in the single cycle graph, for constraint costs sampled from a uniform distribution. Each sub-graph corresponds to a problem with a different domain size (stated at the top of each graph).

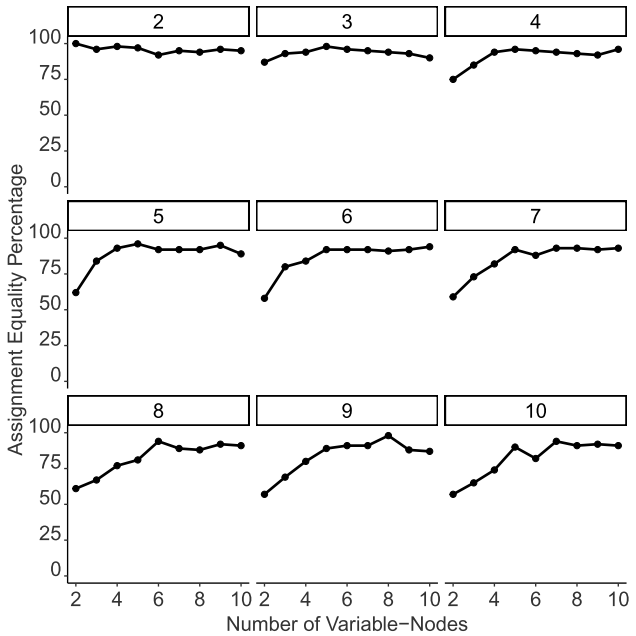



**Fig. 17.** Number of instances among the 100 that did not converge in which assignment equality was generated as a function of the number of variable-nodes in the single cycle graph, for constraint costs taken from a uniform distribution. Each sub-graph corresponds to a different domain size.

## 6. 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. To avoid belief equality, Farinelli et al. [4] proposed the unary constraints method, in which every possible value assignment incurs a very small, randomly-selected added cost. The purpose of invoking this method is to reduce the probability of ties to a negligible level.

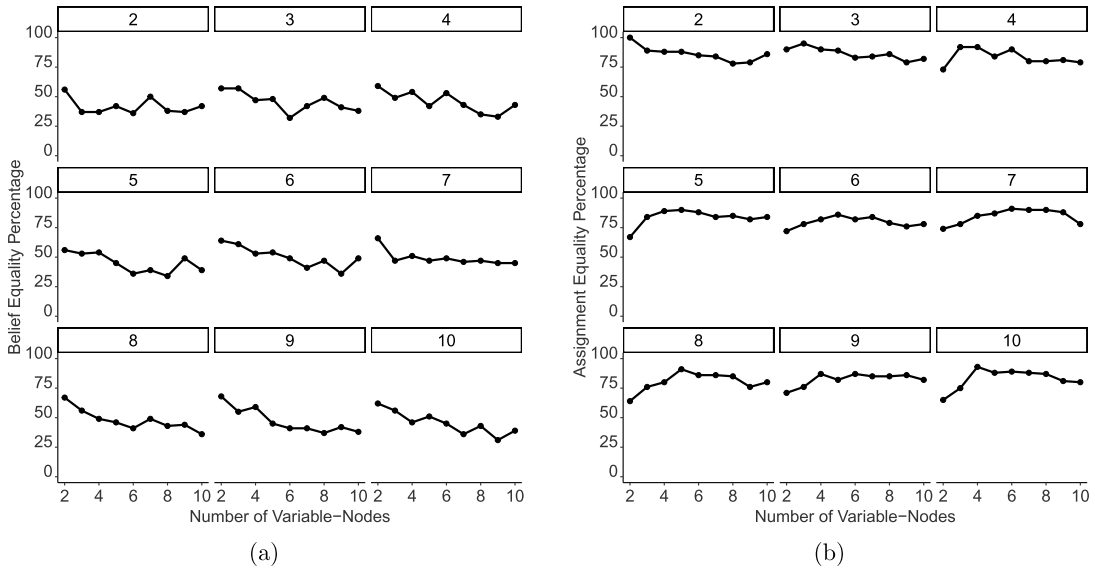



**Fig. 18.** Number of instances, among the 100 that did not converge, in which (a) belief equality and (b) assignment equality were generated as a function of the number of variable-nodes in the single cycle graph, for problems with constraint costs sampled from a Poisson distribution. Each sub-graph corresponds to a different domain size.

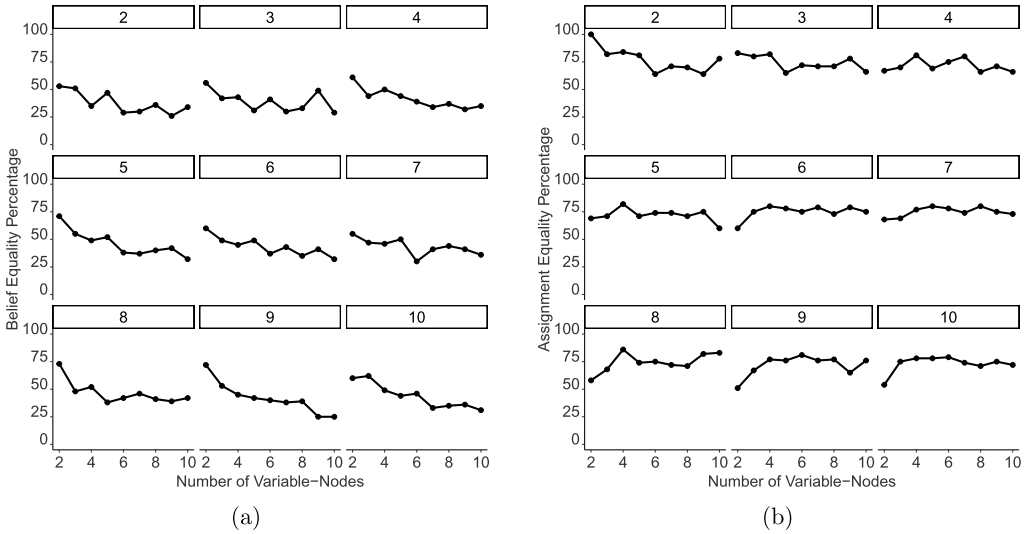



**Fig. 19.** Number of instances, among the 100 that did not converge, in which (a) belief equality and (b) assignment equality were generated as a function of the number of variable-nodes in the single cycle graph, for problems with constraint costs sampled from an index-based Poisson distribution. Each sub-graph corresponds to a different domain size.

When the algorithm converges, such as in the case where it solves a tree-structured graph, Farinelli et al.'s approach for avoiding belief equality does indeed work. However, we prove that even when graphs include just a single cycle, this method cannot prevent belief and assignment equalities. Thus, *ties cannot be avoided*, and our results establish conditions for such equalities in a graph with a single cycle. We suspect that this phenomenon would occur 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.

This work opens up a wide field of research. First, existing results must be re-examined to determine how they are affected by the existence of ties. Second, additional, novel methods should be developed for avoiding ties. In our future research, we plan to investigate whether the results presented in this paper also apply when methods that are known to substantially improve the performance of min-sum on graphs with multiple cycles, such as damping and splitting, are used.

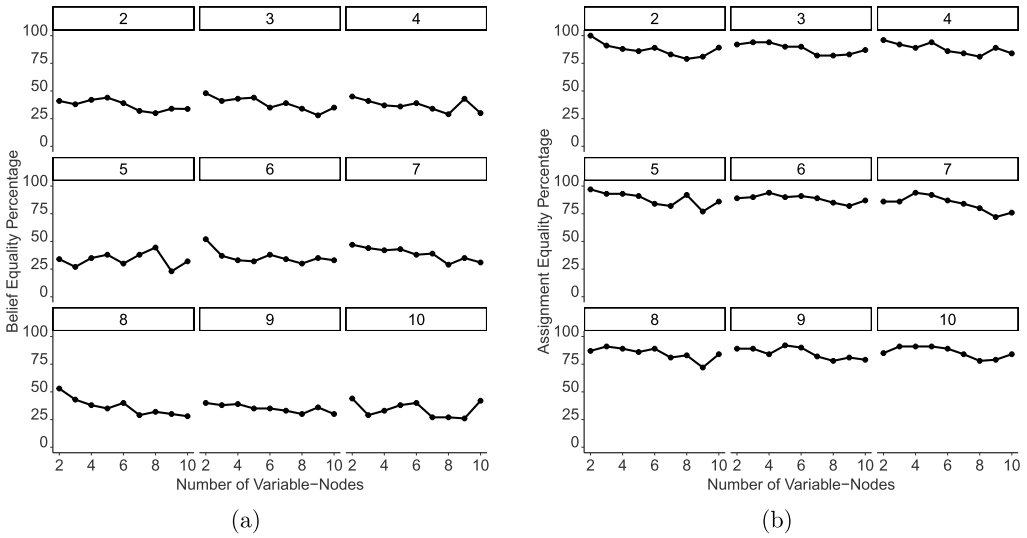



**Fig. 20.** Number of instances, among the 100 that did not converge, in which (a) belief equality and (b) assignment equality were generated as a function of the number of variable-nodes in the single cycle graph, for problems with a single preferred value in each domain. Each sub-graph corresponds to a different domain size.

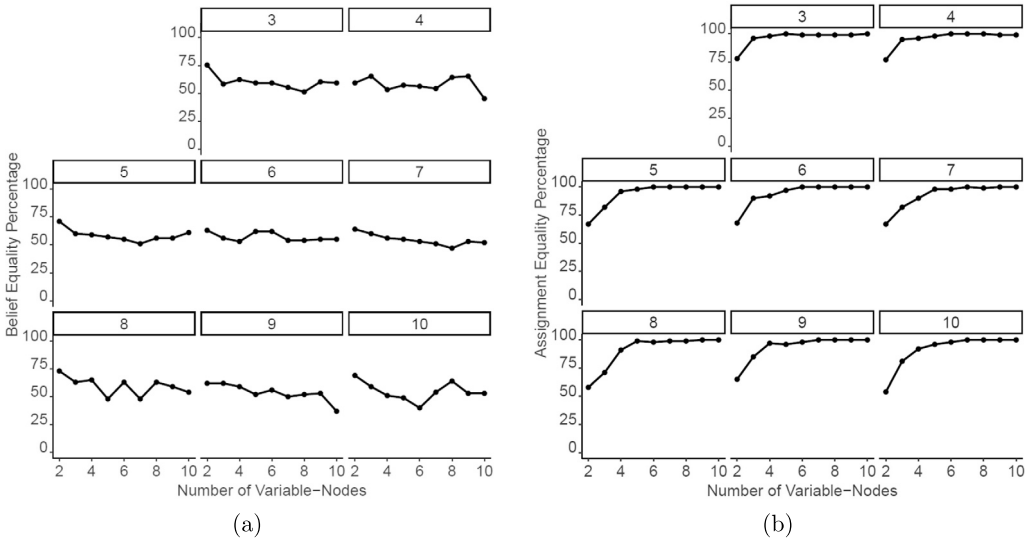



**Fig. 21.** Number of instances, among the 100 that did not converge, in which (a) belief equality and (b) assignment equality were generated as a function of the number of variable-nodes in the single cycle graph, for problems with three preferred values in each domain. Each sub-graph corresponds to a different domain size. Notice, we did not perform experiments with domain size 2, since the amount of preferred values cannot exceed the domain size.

## CRediT authorship contribution statement

**Erel Cohen:** Writing – original draft, Software, Investigation. **Ben Rachmut:** Visualization, Software, Investigation. **Omer Lev:** Writing – review & editing, Writing – original draft, Validation, Investigation, Formal analysis, Conceptualization. **Roie Zivan:** Writing – review & editing, Writing – original draft, Validation, Methodology, Investigation, Formal analysis, Conceptualization.

## Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: Roie Zivan, Ben Rachmut and Erel Cohen report financial support was provided by Binational Science Foundation grant #2018081. Omer Lev reports financial support was provided by Israel Science Foundation grant #196520.

## Appendix A. Equality results

Figs. 18, 19, 20 and 21 present similar results to those depicted in Figs. 16 and 17 when solving problems with the other types of constraint sampling distribution. It seems that the nature of the constraint-cost distribution does not affect the trends observed with regard to the generation of belief and assignment equalities.

## Data availability

Data will be made available on request.

## References

[1] Ziyu Chen, Yanchen Deng, Tengfei Wu, Zhongshi He, A class of iterative refined max-sum algorithms via non-consecutive value propagation strategies, Auton. Agents Multi-Agent Syst. 32 (6) (2018) 822–860.
[2] Liel Cohen, Rotem Galiki, Roie Zivan, Governing convergence of max-sum on dcops through damping and splitting, Artif. Intell. 279 (2020).
[3] Rina Dechter, Constraint Processing, Morgan Kaufman, 2003.
[4] Alessandro Farinelli, Alex Rogers, Adrian Petcu, Nicholas R. Jennings, Decentralised coordination of low-power embedded devices using the max-sum algorithm, in: Proceeding of the 7th International Conference on Autonomous Agents and Multi-agent Systems (AAMAS), 2008, pp. 639–646.
[5] Alessandro Farinelli, Alex Rogers, Nick R. Jennings, Agent-based decentralised coordination for sensor networks using the max-sum algorithm, Auton. Agents Multi-Agent Syst. 28 (3) (2014) 337–380.
[6] G. David Forney, Frank R. Kschischang, Brian Marcus, Selim Tuncel, Iterative decoding of tail-biting trellises and connections with symbolic dynamics, in: Brian Marcus, Joachim Rosenthal (Eds.), Codes, Systems, and Graphical Models, Springer, 2001, pp. 239–264.
[7] Tom Heskes, On the uniqueness of loopy belief propagation fixed points, Neural Comput. 16 (11) (2004) 2379–2413.
[8] Alexander T. Ihler, John W. Fisher III, Alan S. Willsky, Loopy belief propagation: Convergence and effects of message errors, J. Mach. Learn. Res. 6 (2005) 905–936.
[9] F.R. Kschischang, B.J. Frey, H.A. Loeliger, Factor graphs and the sum-product algorithm, IEEE Trans. Inf. Theory 47 (2) (2001) 181–208.
[10] Radu Marinescu, Rina Dechter, AND/OR branch-and-bound search for combinatorial optimization in graphical models, Artif. Intell. 173 (16–17) (2009) 1457–1491.
[11] J. Pearl, Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference, Morgan Kaufmann, San Francisco, California, 1988.
[12] A. Petcu, B. Faltings, A scalable method for multiagent constraint optimization, in: IJCAI, 2005, pp. 266–271.
[13] S.D. Ramchurn, A. Farinelli, K.S. Macarthur, N.R. Jennings, Decentralized coordination in robocup rescue, Comput. J. 53 (9) (2010) 1447–1461.
[14] A. Rogers, A. Farinelli, R. Stranders, N.R. Jennings, Bounded approximate decentralized coordination via the max-sum algorithm, Artif. Intell. 175 (2) (2011) 730–759.
[15] Nicholas Ruozzi, Sekhar Tatikonda, Message-passing algorithms: Reparameterizations and splittings, IEEE Trans. Inf. Theory 59 (9) (2013) 5860–5881.
[16] Ruben Stranders, Alessandro Farinelli, Alex Rogers, Nicholas R. Jennings, 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, 2009, pp. 299–304.
[17] Sekhar Tatikonda, Michael I. Jordan, Loopy belief propagation and Gibbs measures, in: Adnan Darwiche, Nir Friedman (Eds.), UAI '02, Proceedings of the 18th Conference in Uncertainty in Artificial Intelligence, University of Alberta, Edmonton, Alberta, Canada, August 1-4, 2002, Morgan Kaufmann, 2002, pp. 493–500.
[18] W.T.L. Teacy, A. Farinelli, N.J. Grabham, P. Padhy, A. Rogers, N.R. Jennings, Max-sum decentralized coordination for sensor systems, in: AAMAS, 2008, pp. 1697–1698.
[19] Yair Weiss, Correctness of local probability propagation in graphical models with loops, Neural Comput. 12 (1) (2000) 1–41.
[20] Yair Weiss, William T. Freeman, On the optimality of solutions of the max-product belief-propagation algorithm in arbitrary graphs, IEEE Trans. Inf. Theory 47 (2) (2001) 736–744.
[21] C. Yanover, T. Meltzer, Y. Weiss, Linear programming relaxations and belief propagation - an empirical study, J. Mach. Learn. Res. 7 (2006) 1887–1907.
[22] Roie Zivan, Tomer Parash, Liel Cohen, Hilla Peled, Steven Okamoto, Balancing exploration and exploitation in incomplete min/max-sum inference for distributed constraint optimization, Auton. Agents Multi-Agent Syst. 31 (5) (2017) 1165–1207.
[23] Roie Zivan, Omer Lev, Rotem Galiki, Beyond trees: Analysis and convergence of belief propagation in graphs with multiple cycles, in: Proceedings of the 34th Conference on Artificial Intelligence (AAAI), New York City, New York, February 2020, pp. 7333–7340.