Generalized Survey Propagation

by

Ronghui Tu

Thesis submitted to the

Faculty of Graduate and Postdoctoral Studies

In fulfillment of the requirements

For the Ph.D. degree in

Computer Science

Ottawa-Carleton Institute of Computer Science

School of Information Technology and Engineering

University of Ottawa

Abstract

Survey propagation (SP) has recently been discovered as an efficient algorithm in solving classes of hard constraint-satisfaction problems (CSP). Powerful as it is, SP is still a heuristic algorithm, and further understanding its algorithmic nature, improving its effectiveness and extending its applicability are highly desirable.

Prior to the work in this thesis, Maneva et al. introduced a Markov Random Field (MRF) formalism for k-SAT problems, on which SP may be viewed as a special case of the well-known belief propagation (BP) algorithm. This result had sometimes been interpreted to an understanding that "SP is BP" and allows a rigorous extension of SP to a "weighted" version, or a family of algorithms, for k-SAT problems.

SP has also been generalized, in a non-weighted fashion, for solving non-binary CSPs. Such generalization is however presented using statistical physics language and somewhat difficult to access by more general audience.

This thesis generalizes SP both in terms of its applicability to non-binary problems and in terms of introducing "weights" and extending SP to a family of algorithms. Under a generic formulation of CSPs, we first present an understanding of non-weighted SP for arbitrary CSPs in terms of "probabilistic token passing" (PTP). We then show that this probabilistic interpretation of non-weighted SP makes it naturally generalizable to a weighted version, which we call weighted PTP.

Another main contribution of this thesis is a disproof of the folk belief that "SP is BP". We show that the fact that SP is a special case of BP for k-SAT problems is rather incidental. For more general CSPs, SP and generalized SP do not reduce from BP. We also established the conditions under which generalized SP may reduce as special cases of BP.

To explore the benefit of generalizing SP to a wide family and for arbitrary, particu-

larly non-binary, problems, we devised a simple weighted PTP based algorithm for solving 3-COL problems. Experimental results, compared against an existing non-weighted SP based algorithm, reveal the potential performance gain that generalized SP may bring.

献给我亲爱的爸爸妈妈

Acknowledgements

I would like to thank my thesis supervisor Dr Yongyi Mao for opening my research perspectives, for making me appreciate and pursue research of true scientific values, for guiding me through this thesis project, and for spending significant amount of time correcting my writings.

I would also like to thank my thesis supervisor Dr Jiying Zhao for his consistent support during my PhD program and for putting up with my distractions from time to time.

I would like to thank my colleagues and friends at the University of Ottawa, the interactions with whom have made my PhD program a very enjoyable experience.

Finally, I would like thank my dear parents for their support, encouragement, and care, without which this thesis would not have been possible.

All errors in this thesis are mine.

Contents

List of Figures

Chapter 1

Introduction

Survey propagation (SP) [28] is a recent algorithmic breakthrough in solving certain hard families of constraint satisfaction problems (CSPs). Derived from statistical physics, SP first demonstrated its power in solving classic prototypical NP-complete problems, the k-SAT problems [13]. — For random instances of these problems in the hard regime, SP is shown to be the first efficient solver [28]. Recently, SP has also been applied to other CSPs, including other NP-complete problem families such as graph coloring (or q-COL) problems [10], as well as problems arising in communications and data compressions, some examples being coding for Blackwell channels [42] and quantization of Bernoulli sequences [40]. In all these cases, great successes have been demonstrated.

Powerful as it appears, SP however largely remains as a heuristic algorithm to date, where analytic understanding of its algorithmic nature and rigorous characterization of its performance are widely open and of great curiosity and research importance.

This thesis aims at developing theoretical insights in understanding SP, generalizing SP from sporadic examples as mentioned above to more general context and with more general forms, and exploring the potential benefits of such generalization.

1.1 Brief Background

1.1.1 "SP is BP"

Prior to this work, a connection between SP and a well-known algorithm used in iterative decoding [33] and statistical inference [32], Belief propagation (BP), was observed. Similar to BP, SP operates by iteratively passing "messages" in a factor graph representation [21] of the problem instance, where each variable vertex corresponds to a variable whose value is to be decided and each function vertex corresponds to a local constraint imposed on the variables. This observation has inspired a recent research effort in understanding whether SP may be viewed as a special case of BP. — The significance of questions of such a kind has been witnessed repeatedly in the history of communication research, for example, in understanding the Viterbi algorithm as a dynamic programming algorithm [16], in understanding the turbo decoding algorithm [7] as an instance of Belief Propagation [24], and in unifying the BCJR algorithm [6] and the Viterbi algorithm under the umbrella of the generalized distributive law [4], etc. These unified frameworks have on one hand provided additional insights into the nature of the algorithms, and on the other hand allowed an easier access of the algorithm by much wider research communities. Specific to the question "is SP BP?", if SP may be understood as an instance of BP, then the existing analytic techniques of BP are readily applicable to analyzing SP; if SP can not be characterized as a special case of BP, one is then motivated to seek a different algorithmic framework to which SP belongs or to discover the unique algorithmic nature of SP.

The first result reporting that SP may be understood as an instance of BP is the work of [11] in the context of k-SAT problems. This result is generalized in [23] to an extended version of SP for solving k-SAT problems. Briefly, the authors of [23] present a

Markov Random Field (MRF) [19] formalism for k-SAT problems; a parameter, denoted by ϵ in this thesis, is used to parametrize the MRF. When the BP algorithm is derived on such an MRF, the BP message-update equations result in a *family* of SP algorithms, referred to as *weighted SP*[1] or SP(ϵ) in this thesis, parametrized by $\epsilon \in [0, 1]$; and when $\epsilon = 1$, SP(ϵ) is the original (non-weighted) SP. In addition to extending SP — in the context of k-SAT problems — to a family of SP algorithms with tunable performance, another significance of this result is a conclusion that SP is BP for the k-SAT problem family.

Due to these results, the research communities had tended to regard SP (and weighted SP) as a special case of BP prior to the work in this thesis.

1.1.2 Generalizations of SP

As noted earlier, (non-weighted) SP was first introduced in the context of a binary CSP family, the k-SAT problems [28]. It is rather straightforward to apply the same design philosophy to other binary CSPs of similar kinds and to derive the corresponding (non-weighted) SP algorithm. Prior to this work, generalizing SP on two distinct dimensions have been explored.

The first dimension is to extend non-weighted SP from binary problems to non-binary or more general CSP families. Such generalization primarily exists in the literature of statistical physics (see, e.g., [8]). The other dimension is to introduce "weights" in SP update equations. We note that prior to this work, such a generalization has only been presented for k-SAT problems as in [23] and in sporadic example applications involving only *binary* variables such as in [40]. In fact, the design philosophy of weighted SP for CSPs involving binary variables (such as in [23] and [40]) is not readily extendable to

[1] In [23], weighted SP is referred to as generalized SP. We however prefer calling it weighted SP (or SP(ϵ)) instead since a further generalization is introduced in this thesis.

arbitrary CSPs with arbitrary variable alphabets for two reasons:

- First, an important notion underlying SP, namely, an *appropriate* extension of variable alphabets, is blurred in formulation of binary special case in the generalization in [23], although such notion had been brought to surface in the somewhat distant field of statistical physics [8].

- Further complicating the issue is the earlier question whether "SP is BP". The existing generalization of SP to a weighted version for k-SAT problems is well justified when in that context SP is viewed as an instance of BP. However, for arbitrary CSPs, there exists no rigorous understanding concerning the relationship between SP and BP. As such, for arbitrary CSPs, additional principles need to be established for such generalization.

1.2 Thesis Contributions

Towards a rigorous understanding of SP, this research, in retrospect, was initiated by the question whether SP and more generally weighted SP are truly special cases of BP for arbitrary CSPs beyond k-SAT problems, where along the way, we generalize SP on both dimensions, namely, extending it to non-binary problems and introducing weights to the update equations.

The main contributions and results of the thesis are summarized as follows.

1. **Generalizing SP to a weighted version on arbitrary CSP families**

   We formulate SP and weighted SP for general CSPs as what we call "probabilistic token passing" (PTP) and "weighted probabilistic token passing" (weighted PTP) respectively, where a message is a distribution (or non-negative function) on the

set of "tokens" associated with a variable. Here a "token" is a non-empty *subset* of the variable's alphabet[2]. It has been previously observed in SP applied to various problems that a "joker" symbol is added to the original variable alphabet. Here we point out that extending the alphabet by simply adding a joker symbol is not sufficient for general CSPs, particularly for those involving non-binary variables. We stress that the *right* extension of the variable alphabet is to replace it with the set of all non-empty subsets of the original alphabet. Although an equivalent treatment has been described in some previous literature for non-weighted SP [8], this perspective is for the first time made explicit beyond statistical physics context and for both non-weighted and weighted SP. Based on this notion of alphabet extension, we generalize weighted SP for arbitrary CSPs in the form of weighted PTP. In other words, the weighted PTP formulation presented in this work serves as a recipe for designing weighted SP algorithm for arbitrary CSPs.

We develop several analytic results concerning the dynamics of PTP, which may provide intuitions on how PTP and weighted PTP work. These results are however still preliminary and the dynamics of SP algorithms remain largely open to date.

2. **Proving that SP is not BP for general CSPs**

We present an MRF formalism — which we refer to as "normally realized MRF" — for arbitrary CSPs using Forney graphs, generalizing the MRF construction in the style of [23] presented for k-SAT problems. States, each consisting of a left state and a right state, are introduced in the MRF, where the left state corresponds to

[2]In fact more rigorously, as will be shown later in this thesis, a token is a non-empty *subset* of all possible *assignments* of a variable – In this work, for more mathematical rigor and clarity, we make a distinction between the alphabet of a variable and the set of all assignments to the variable, where an assignment to variable x_v is treated as a function mapping the singleton set $\{v\}$ to the alphabet of x_v. Nevertheless, one may always identify the set of all assignments to x_v with the alphabet of x_v via a one-to-one correspondence and loosely refer to the set of all assignments of a variable as the alphabet of the variable.

the token passed from the variable and the right state corresponds to the token passed from the constraint. For any given CSP, the MRF is parametrized by a collection of weighting functions, each corresponding to a variable in the CSP; in the k-SAT special case, these weighting functions reduce to a single parameter.

On the normally realized MRF formalism, we then proceed to derive the BP update equations and investigate the reduction of BP to weighted PTP (noting that weighted PTP *is* weighted SP and that non-weighted SP is a special case of weighted SP). Primarily re-developing the results of [23] and [34] on BP-to-SP reduction, we show that for k-SAT problems, BP is readily reducible to weighted PTP as long as a condition — which we refer to as the *state-decoupling condition* — is imposed on the BP messages in initialization. An interesting fact about this condition in the context of k-SAT problems is that as long as the condition is satisfied in the first BP iteration, it will continue to be satisfied in all iterations after. This forms the basis on which BP messages may be simplified to the form of weighted PTP messages. This condition, also arising in [23] as a peculiar and curious construction, had not been explained prior to this work. In this work, we argue that the state-decoupling condition serves a critical role in the reduction of the weighted PTP messages from the BP messages derived from the MRF formalism in the style of [23] or from the normally realized MRF presented in this work. Using the example of 3-COL problems, we show that such a condition is also needed in all BP iterations so as for BP to reduce to PTP (or SP). However, in that case, we show that this condition can not be made satisfied in every BP iteration (except for the trivial cases in which the BP messages contain no useful information) and one must manually impose this condition by manipulating the BP messages in each iteration. This result on one hand justifies the important role of the state-decoupling condition in the reduction

of BP to PTP and on the other hand asserts that BP is *not* PTP and hence *not* SP for 3-COL problems!

At that point, one is ready to conclude that weighted PTP or weighted SP is not a special case of BP for general CSPs. The manual manipulation of BP messages in 3-COL problems, which results in what we call *state-decoupled BP* brings up a further question, namely, for general CSPs, whether PTP and weighted PTP are readily expressed as state-decoupled BP. We proceed to show that for general CSPs, the reduction of weighted PTP from state-decoupled BP requires yet another condition pertaining to the structure of the CSP. Briefly, this additional condition demands that the constraints in the CSP be "locally compatible" with each other in some sense. We show that the local compatibility condition of the CSP is the necessary and sufficient condition for state-decoupled BP to reduce to weighted PTP or weighted SP. At that end, we complete the answer to the question "is SP BP?".

3. **Demonstrating the advantage of the generalized survey propagation in solving graph coloring problems**

   It is known that similar to the k-SAT problems, 3-COL problems are also NP-complete problems in general. It has been previously estimated that when the average vertex degree θ is within (approximately) the range $[4.42, 4.69]$, a random 3-COL problem is almost surely solvable but difficult [10]. No efficient solvers were proposed for 3-COL problems in this "hard regime" until [10]. In [10], the authors developed a non-weighted-SP-based algorithm, which we refer to as the BMPWZ algorithm. The algorithm incorporates a sophisticated decimation and a local search procedure, and is capable of solving a good fraction of 3-COL problems in the hard regime, although problems in this hard regime with higher average

vertex degrees still can not be solved by the BMPWZ algorithm.

In order to explore the potential benefit of generalized survey propagation introduced in this thesis, we devised a weighted-PTP-based algorithm for random 3-COL problems in the hard regime. Utilizing a simple decimation rule and without a local search procedure, we show that our algorithm already outperforms the BMPWZ algorithm for problems with high average vertex degrees, although for problems with low average vertex degrees, the performance of our algorithm is upper-bounded by a success rate of about 0.9 due to our over-simplified approaches to decimation and local search. Nevertheless, this suggests that generalized survey propagation can indeed bring additional performance gain for solving CSPs.

The following conference and journal papers result from the work presented in this thesis.

- [39] R. Tu, Y. Mao, and J. Zhao, "Is SP BP?" IEEE Transactions on Information Theory, June 2010.
- [38] R. Tu, Y. Mao, and J. Zhao, "Survey propagation as "probabilistic token passing"," IEICE Transactions on Information and Systems, Special Section on Foundations of Computer Science, Feburary 2008.
- [36] R. Tu, Y. Mao, and J. Zhao, "Is SP BP?" in the 2007 IEEE Information Theory workshop, invited paper, 2007.
- [37] R. Tu, Y. Mao, and J. Zhao, "On the interpretation of survey propagation," in the Proceedings of the 10th Canadian Workshop on Information Theory, 2007.
- [34] R. Tu, Y. Mao, and J. Zhao, "On generalized survey propagation: normal realization and sum-product interpretation," In the Proceedings of IEEE International Symposium on Information Theory, 2006.

- [35] R. Tu, Y. Mao, and J. Zhao, "Towards a unified solution for constraint satisfaction problems: A survey-propagation approach based on normal realizations," in the Proceedings of the 23rd Biennial Symposium on Communication, 2006.

Another paper presenting our algorithm for solving 3-COL problems is currently under preparation.

1.3 Thesis Organization

The remainder of this thesis is organized as follows.

Chapter 2 is a concise tutorial overview of factor graphs, belief propagation and related algorithms.

In Chapter 3, we present a generic formulation of constraint satisfaction problems, followed by a review of the existing SP algorithms, where we focus on the examples of the k-SAT and 3-COL problem families. A lemma is proved in this chapter, which provides an alternative but equivalent formulation of weighted SP for k-SAT problem.

We then proceed in Chapter 4 to present a generalized version of survey propagation, weighted PTP (probabilistic token passing). Using the examples of k-SAT and 3-COL problems, we show that weighted PTP generalizes the existing SP algorithms. Analytic results and discussions on the dynamics of SP and generalized SP are also provided in Chapter 4.

Chapter 5 studies the connection between generalized SP (namely weighted PTP) and BP. We show that SP is in general not BP and derived the conditions under which SP may be regarded as an instance of BP.

In Chapter 6, we present an algorithm based on generalized survey propagation and a simple decimation rule for solving random 3-COL problems. Although bottlenecked

by the performance of decimation, we show that the algorithm already outperforms the BMPWZ algorithm presented in [8].

Chapter 7 concludes the thesis and points to possible future directions of research which may follow upon this work.

Chapter 2

Factor Graphs and Belief Propagation

In this chapter, we review the notions of factor graphs and a special kind of factor graphs, Forney graphs. We also explain the well-known belief propagation algorithm and one of its close relatives, the max-product algorithm. We begin by introducing some notation, which will be used throughout the thesis.

2.1 Notation

Let V be a finite set, in which each element will be referred to as a *coordinate*. Associated with each $v \in V$, there is a finite *alphabet* χ_v. For each $v \in V$, we will assume throughout this thesis that every χ_v is identical to each other, and is therefore denoted by χ. We note that this slight loss of generality is made only for lightening the upcoming notations, and that there is no difficulty to extend the results in this thesis to more general cases where χ_v's are different from each other. For any subset $U \subseteq V$, a *χ-assignment* x_U on U is a function mapping U into the set χ. That is, a χ-assignment x_U specifies a way

to assign each coordinate $u \in U$ a value in χ. The set of all χ-assignments on U will be denoted by χ^U. When U is a singleton set $\{u\}$, which contains a single coordinate u, we will call χ-assignment $x_{\{u\}}$ on $\{u\}$ an *elementary (χ-)assignment* and write it as x_u for simplicity. Clearly, any given elementary χ-assignment x_u is uniquely specified by a value $r \in \chi$, which is the assigned value in χ to coordinate u. In this case, this assignment is denoted by r_u, for example, if $\chi := \{0, 1\}$, then the only possible χ-assignments on $\{u\}$ are 0_u and 1_u, which are the elementary assignments assigning 0 and 1 to coordinate u, respectively.

Suppose that $U \subset W \subseteq V$ and that x_W is a χ-assignment on W. We will use $x_{W:U}$ to denote the (function) restriction of x_W on U. For any subset of χ-assignments $\Omega \subseteq \chi^W$ on W, we denote the projection of Ω on U by $\Omega_{:U}$. That is,

$$\Omega_{:U} := \{x_{W:U} | x_W \in \Omega\}.$$

If coordinate set U can be partitioned into disjoint subsets A and B, then it is obvious that assignment x_U decomposes into assignments $x_{U:A}$ and $x_{U:B}$, and x_U may be written as $(x_{U:A}, x_{U:B})$ (in any order). Evidently, x_U may be decomposed according to any partition of U, not necessarily two-fold partitions. In particular, if a collection of sets $\{U_i | i \in \mathcal{I}\}$, for some $\mathcal{I}$, form a partition of U, then we may write assignment x_U as $\langle x_{U:U_i} \rangle_{i \in \mathcal{I}}$.

For simplicity, we will write (x_A, x_B) and $\langle x_{U_i} \rangle_{i \in \mathcal{I}}$ in place of $(x_{U:A}, x_{U:B})$ and $\langle x_{U:U_i} \rangle_{i \in \mathcal{I}}$ respectively. In fact, unless some particular clarity is needed, we will always write $x_{W:U}$ simply as x_U, making the underlying x_W implicit. Furthermore, when U is a singleton set $\{u\}$, as mentioned earlier, we will simply denote it by x_u, which reduces to the conventional "variable" notation in standard literatures of graphical models.

2.2 Factor graphs

Let C be a finite set indexing a set of real-valued functions $\{f_c : c \in C\}$, where each function f_c, $c \in C$, applies to a subset $V(c)$ of coordinate V. More specifically, function f_c, $c \in C$, is a mapping from $\chi^{V(c)}$ to $\mathbb{R}$, the set of all real numbers.

Globally a factor graph represents the product $\prod_{c \in C} f_c(x_{V(c)})$.

A factor graph G is a bipartite graph consisting of a set of "variable vertices" representing the set of variables $\{x_v : v \in V\}$ and a set of function vertices representing functions $\{f_c : c \in C\}$. An edge connects a variable vertex x_v and a function vertex f_c if and only if $v \in V(c)$. Clearly, a factor graph is completely specified via $(V, C, \{V(c) : c \in C\}, \{f_c : c \in C\})$.

Conversely, if any multi-variate function $F : \chi^V \to \mathbb{R}$ factors into the product of local functions according to

$$F(x_V) = \prod_{c \in C} f_c(x_{V:V(c)}),$$

for some choice of $\{f_c : c \in C\}$, then this factorization structure can be expressed by a factor graph.

Example 1 *Suppose* $F(x_1, x_2, x_3, x_4) = f_1(x_1, x_2) \cdot f_2(x_2, x_3, x_4) \cdot f_3(x_2, x_4)$. *Then* F *can be represented by the factor graph shown in Figure 2.1.*

In the framework of factor graphs, most commonly encountered multi-variate functions are constraint-indicator functions and more generally probability distributions.

We will adopt the Iverson's convention [21] to denote a constraint-indicator function: for any boolean proposition, $[P]$ evaluates to 1 if P and to 0 otherwise.

We now give an example of factor graph modeling in the field of error-control coding, where factor graphs and the algorithms therein (namely, Belief Propagation, which will be introduced later this chapter) have revolutionized the field.

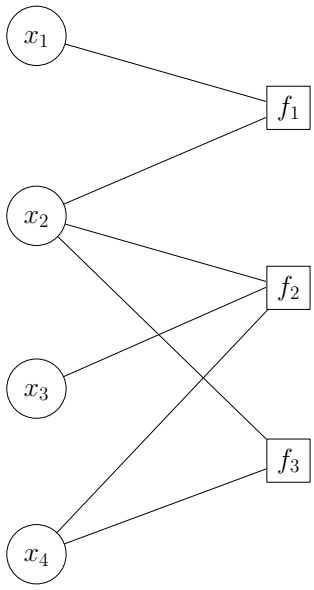


Figure 2.1: A factor graph representing function F in Example 1.

Example 2 *A linear error correcting code can be represented by its parity-check matrix. For example, the well-known* $(7, 4)$ *Hamming code has a parity-check matrix*

$$H = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}.$$

Essentially, parity-matrix H *above specifies that the Hamming code* C *consists of the set of all binary vectors* $\boldsymbol{x} = \{x_1, x_2, x_3, x_4, x_5, x_6, x_7\}$ *that satisfy* $H\boldsymbol{x}^T = 0$*. In other words, the codewords should satisfy* $[x_4{+}x_5{+}x_6{+}x_7 = 0]\cdot[x_2{+}x_3{+}x_6{+}x_7 = 0]\cdot[x_1{+}x_3{+}x_5{+}x_7 = 0] = 1$.

Let $f_1(x_1, x_3, x_5, x_7) := [x_1{+}x_3{+}x_5{+}x_7 = 0]$*,* $f_2(x_2, x_3, x_6, x_7) := [x_2{+}x_3{+}x_6{+}x_7 = 0]$*, and* $f_3(x_4, x_5, x_6, x_7) := [x_4 + x_5 + x_6 + x_7 = 0]$*. Then the product of local functions* f_1*,* f_2 *and* f_3 *can be represented by the factor graph in Figure 2.2.*

An indicator function for a constraint may be viewed as, up to scale, the uniform

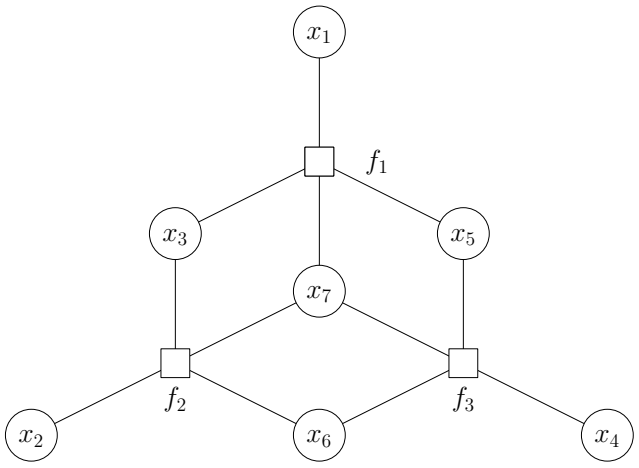
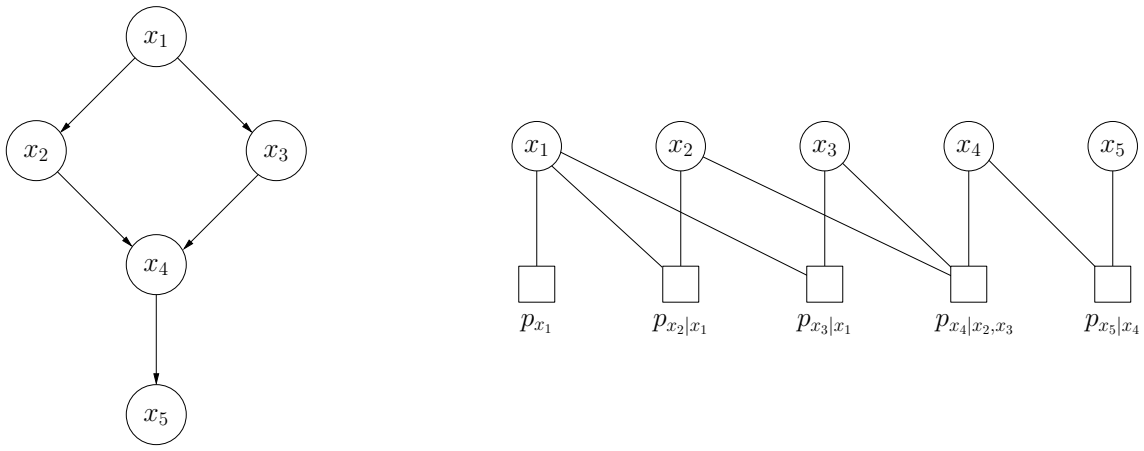


Figure 2.2: A factor graph model for $(7, 4)$ Hamming code based on the parity check matrix in Example 2.

probability distribution over the set of configurations satisfying the constraint. Indeed, factor graphs are often used to model distributions in more general forms rather than uniform distributions. In typical distributions involving a large number of random variables, the interactions of the random variables are usually sparse. This fact makes factor graphs an appealing model for systems of large number of variables, since as we will show via the following example, the interaction topology of the random variables can be made corresponding to the structure of the factor graph.

Example 3 *Figure 2.3 (a) is a Bayesian network [32] showing that joint probability distribution* $p(x_1, x_2, x_3, x_4, x_5)$ *can be factorized as* $p(x_1, x_2, x_3, x_4, x_5) = p(x_1) \cdot p(x_2|x_1) \cdot p(x_3|x_1) \cdot p(x_4|x_2, x_3) \cdot p(x_5|x_4)$. *Figure 2.3 (b) is the factor graph representing this factorization structure.*

In the above example, each factor consisting of the global distribution is either a conditional distribution or a marginal distribution. This need not to be the case in general when a factor graph is used as the probability model. Specifically, when the random variables interact with "equal footing" rather than in a certain causal relationship, a



(a) (b)

Figure 2.3: (a) A Bayesian network that shows the joint probability distribution $p(x_1, x_2, x_3, x_4, x_5) = p(x_1) \cdot p(x_2|x_1) \cdot p(x_3|x_1) \cdot p(x_4|x_2, x_3) \cdot p(x_5|x_4)$. (b) A factor graph that represents the factorization shown in (a).

Bayesian network representation of the joint distribution does not necessarily exist, but the factor graph model may still be applicable. Example 2 may be regarded as such a case when the global constraint indicator function is treated as a uniform distribution over the set of all codewords.

When the global function represented by the factor graph is (up to scale) a probability distribution, namely, probability mass function (PMF) or probability density function (PDF), the factor graph essentially defines what is known as a Markov Random Field (MRF) [19] in the literature. Without giving a rigorous definition of MRF for simplicity, in this thesis we will simply treat factor graphs representing a joint distribution as MRFs, although some slight but non-critical differences exist between MRF and factor-graph probability models. Specifically, in this context, a so-called *global Markov property* holds[1]:

[1]Markov random fields are a special case of probability models using undirected graphs where each vertex represents a random variable. There are several types of Markov property which may axiomatically define various probability models based on undirected graph representations. The most well-known Markov properties are global Markov property, local Markov property, and pair-wise Markov property. Markov random field is originally defined using local Markov property. It is however known that when

for any three sets of variable vertices (x_A, x_S, x_B) where x_S separates x_A from x_B on the factor graph, x_A is independent of x_B given x_S.

2.3 Forney Graphs

A Forney graph, originally introduced by Forney [17] under the name "normal realization", is a special type of factor graph. In a Forney graph, each variable vertex has either degree one (i.e., being a leaf variable vertex) or degree two (i.e., connecting exactly two function vertices). When such a constraint is imposed on the factor graph, it is convenient to suppress the degree-2 variable vertices — which Forney refers to as *(generalized) state variables* — and use edges to represent them. The degree-one variable vertices are represented using "half edges" in Forney graphs.

Forney showed in [17] that any factor graph can be represented alternatively by a Forney graph by introducing additional replica variables and some function vertices representing the equality constraint indicator between a variable and its replicas. This may be seen in the following example.

Example 4 *Figure 2.4 is a Forney graph that represents the same problem shown in Figure 2.1.*

The vertices labeled by symbol "=" are for equality constraints. In this example, we have four equality constraints: $[x_1 = s_{11}]$, $[x_2 = s_{21} = s_{22} = s_{23}]$, $[x_3 = s_{31}]$, *and* $[x_4 = s_{41} = s_{42}]$. *The original local functions now involve the state variables, namely in the form of* $f_1(s_{11}, s_{21})$, $f_2(s_{22}, s_{31}, s_{41})$, *and* $f_3(s_{23}, s_{42})$.

The significance of Forney graph representation can hardly be overstated. Essentially, Forney graphs may be viewed as a generalization of state-space models and a most fun-

the joint distribution factors as the product of the functions each on the clique of the undirected graph, all three types of Markov property are equivalent [22].

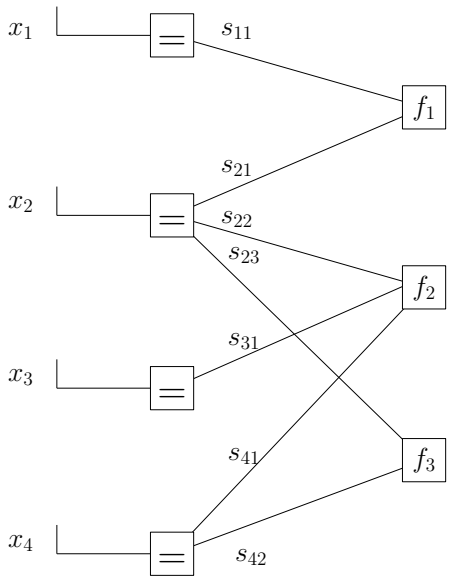


Figure 2.4: A Forney graph representing the same problem shown in Figure 2.1.

damental modern concept in system theory. When Forney graphs obey a chain structure, they reduce to the conventional state-space model; and when the graph obeys arbitrary structure, any cut of the graph (at the edges or states) characterizes the state linking the variable configurations on two sides of the cut.

When modeling error-correcting codes, Forney graphs can be "dualized" via local operation of the function vertices, and the dual Forney graph represents the dual code.

2.4 Function Marginalization

Given a multi-variate function $F(x_V) : \chi^V \to \mathbb{R}$, the *marginal (function)* of $F(x_V)$ on x_v for some $v \in V$ is a function $\chi^{\{v\}} \to \mathbb{R}$ defined as

$$\sum_{x_{V\setminus\{v\}}} F(x_V) \tag{2.1}$$

A common computational task that may arise with a multi-variate function $F(x_V)$ is the computation of its marginal functions. This is particularly the case when $F(x_V)$ is a distribution or a conditional distribution, and such problems typically arise in the context of statistical inference. For example, in statistical inference, one often needs to compute the marginal of $F(x_V)$ on some or even every $v \in V$, since the maximizing assignment $\hat{x}_v$ of the marginal is the mostly likely configuration of x_v under distribution F. That is, a statistical inference problem may often be formulated as finding

$$\hat{x}_v := \arg\max_{x_v} \sum_{x_{V\setminus\{v\}}} F(x_V), \tag{2.2}$$

for every $v \in V$.

In general when F involves a large number of variables, such computational tasks, particularly those which compute the marginal function of F on every v, require computational complexity exponential in number of variables. However, when F factors according to a factor graph that is only sparsely connected, it is possible to perform the computation optimally or sub-optimally with complexity linear in the number of variables, as we will outline in next section.

2.5 Belief Propagation

When multi-variate function $F(x_V)$ is represented by a factor graph, the computational problems specified in (2.2) for all $v \in V$ can be solved using the belief propagation (BP) algorithm, also known as the sum-product algorithm [21].

The BP algorithm is an iterative algorithm operating on the factor graph by passing messages between variable vertices and function vertices. A message is a function of a single variable and is only passed between adjacent vertices in the graph or computed

locally as the "summary message" at each variable vertex. More specifically, the message $\mu_{c\to v}$ passed from a function vertex f_c to an adjacent variable vertex x_v and the message $\mu_{v\to c}$ passed from variable vertex x_v to function vertex f_c are both functions of variable x_v; the summary message μ_v at each variable vertex x_v is also a function of x_v.

The BP message-update rule is given next, where we will assume that variable x_v and function f_c are adjacent in the factor graph. We will use $C(v)$ to denote the index set of function vertices involving variable x_v and $V(c)$ to denote the index set of variable vertices involved in function f_c.

BP Message-Update Rule

$$\mu_{v\to c}(x_v) := \prod_{c'\in C(v)\setminus\{c\}} \mu_{c'\to v}(x_v), \tag{2.3}$$

$$\mu_{c\to v}(x_v) := \sum_{x_{V(c)\setminus\{v\}}} f_c(x_{V(c)}) \prod_{v'\in V(c)\setminus\{v\}} \mu_{v'\to c}(x_{v'}) \tag{2.4}$$

$$\mu_v(x_v) := \prod_{c\in C(v)} \mu_{c\to v}(x_v). \tag{2.5}$$

The update of messages may be arranged in various orders, or "schedules". There is a basic rule in every schedule: a vertex only passes a message to a neighbor when it has available all messages incoming from all other neighbors.

On factor graph without cycles, a commonly used schedule is that message-passing starts from leaf vertices, where if the leaf vertex is a variable vertex x_v, it passes constant function 1 (to its only neighbor) and if the leaf vertex is a function vertex $f_c(x_u)$, it simply passes function $f_c(x_u)$ (to its only neighbor x_u). After this initialization step, the message-update simply follows the basic rule. When along every edge, messages passed to both directions become available, the summary message at each variable vertex is

computed and the algorithm terminates. Due to the cycle-free nature of the factor graph, it is apparent that the termination of BP takes a finite number of message-updates. After BP terminates, it is then possible to show that at every vertex x_v, the summary message μ_v is precisely the marginal of F on x_v. That is, on factor graph without cycles, BP solves the inference problem specified in (2.2).

On a factor graph with cycles, the most commonly used message-update schedule is what is known as the *flooding schedule*, in which the message-updates are packaged in *iterations*. In each iteration, first all the variable vertices pass messages, then all the function vertices pass messages, and the iteration ends with the computation of summary messages at all variable vertices. Prior to the first iteration, the messages to each variable vertex are initialized to the constant function 1. The message-update terminates after a pre-specified number of iterations or upon the convergence of the messages. For factor graphs with cycles, it is no longer the case that the summary messages are the desired marginal functions of F. However, experimental results have demonstrated that when the graph is large and sparse, the maximizing configuration of each summary message is usually the solution or close to the solution of the inference problem specified in (2.2). This fact precisely underlies the recent celebrated success in applying BP to decoding error correcting codes represented by factor graphs.

2.6 Max-Product Algorithm

It is observed that the fundamental principle that allows the BP algorithm to save on computation is the distributive law between addition "+" and multiplication "$\cdot$" [4, 21]. That is, for any $x, y, z \in \mathbb{R}$,

$$x \cdot (y + z) = x \cdot y + x \cdot z.$$

This observation allows a generalization of BP to a family of algorithms, in which each algorithm may possibly serve a different purpose and in which the max-product algorithm is of great relevance to this work.

Let us consider the following distributive law: for any $x, y, z > 0$,

$$x \cdot \max\{y, z\} = \max\{x \cdot y, x \cdot z\}.$$

This distributive law essentially underlies the max-product algorithm which we now explain.

Suppose that a multi-variate real-valued function $F(x_V)$ factors as the product of a set of positive local functions. In many cases (for example in statistical inference), we may be interested in finding a valid configuration that maximizes function $F(x_V)$. In this setting, one may apply the max-product algorithm on the factor graph representing F. The message-update rule of the max-product algorithm is given as follows:

$$\mu_{v\to c}^{\max-\text{prod}}(x_v) := \prod_{c'\in C(v)\setminus\{c\}} \mu_{v\to c'}^{\max-\text{prod}}(x_v) \tag{2.6}$$

$$\mu_{c\to v}^{\max-\text{prod}}(x_v) := \max_{x_{V(c)\setminus\{v\}}} \left(f_c(x_{V(c)}) \prod_{v'\in V(c)\setminus\{v\}} \mu_{v'\to c}^{\max-\text{prod}}(x_{v'}) \right) \tag{2.7}$$

$$\mu_{v}^{\max-\text{prod}}(x_v) := \prod_{c'\in C(v)} \mu_{v\to c'}^{\max-\text{prod}}(x_v) \tag{2.8}$$

where $\mu_{v\to c}^{\max-\text{prod}}$, $\mu_{c\to v}^{\max-\text{prod}}$ and $\mu_{v}^{\max-\text{prod}}$, all functions on χ, are respectively a left message (i.e., message passed from variable x_v to function f_c), a right message (i.e., message passed from function f_c to variable x_v), and a summary message (i.e., message computed at variable x_v).

If the factor graph representing F is cycle-free, then using a similar message-passing schedule as that of BP, it is possible to show that upon convergence, the values of all variables each maximizing its summary message form the configuration that maximizes F globally. The

algorithm may also be applied to factor graphs with cycles, in which case the obtained global configuration approximates the true maximizing configuration of F.

2.7 The Equivalence of Message-Passing Algorithms

Much of the work in this thesis has to do with the "equivalence" between message-passing algorithms. Here we take a slight digression to explain this notion of equivalence, which will be used throughout the thesis.

Abstractly, the algorithms we consider may all be described in terms of the following pseudo-code.

```
Initialize Q;
While(looping condition satisfied){
    Q_new := f(Q_old)
}
R := g(Q)
```

In essence, each message-passing algorithm iteratively updates a vector-valued quantity Q according to some function f, and eventually outputs a vector-valued quantity R from the resulting Q according to some function g. In this setting, two algorithms A and B are said to equivalent if they use the same function f, namely, $f^A = f^B$. That is, the equivalence we consider is the equivalence between the rules that update the respective quantities Q^A and Q^B.

Such an equivalence can always be established by identifying the components of Q^A with the components of Q^B. If the i^{th} component $Q^A(i)$ of Q^A is identified with the j^{th} component $Q^B(j)$ of Q^B, we write $Q^A(i) \leftrightarrow Q^B(j)$.

The proofs of algorithm equivalence in this thesis are solely established by such a correspondence between the components of Q's. We however make no effort to explicitly define Q and f in our proofs, in order to simplify the proofs. This shall result in no ambiguity and, in fact, better clarity.

We note that this notion of equivalence only deals with the equivalence between the update rules of the compared algorithms. In fact, since in most cases, the function g is closely related to the function f, such an equivalence may also be extended to the equivalence between the rules of computing the respective outputs, R's.

2.8 Concluding Remarks

This chapter reviews the framework of factor graphs, belief propagation and related algorithms. We single out the notion of Forney graphs as a special kind of factor graphs, and will use this graphical model extensively in the rest of the thesis. We provide a self-contained exposition of the belief propagation and max-product algorithms, both of which will appear relevant to the development of this thesis.

Chapter 3

Constraint Satisfaction Problems and Survey Propagation

In this chapter, we will present a generic formulation of constraint satisfaction problems (CSPs), sufficiently general for arbitrary CSPs. We will also review existing SP algorithms developed for several classes of CSPs and present some elementary results.

3.1 A Generic Formulation of Constraint Satisfaction Problems

Given variable alphabet χ and index set V, the objective of a constraint satisfaction problem (CSP) is to find a global χ-assignment x_V that satisfies a given set of constraints or to conclude that no such assignment exists. Formally, we will use set C to index the set of constraints $\{\Gamma_c : c \in C\}$. Each constraint Γ_c, $c \in C$, applies to a subset of the coordinates V, which will be denoted by $V(c)$. Specifically, each constraint Γ_c is identified with a subset of $\chi^{V(c)}$, and the constraint is satisfied by global χ-assignment x_V if $x_{V:V(c)} \in \Gamma_c$. Then any CSP may be formulated via specifying V, C, χ, $\{V(c) : c \in C\}$ and $\{\Gamma_c : c \in C\}$, where the objective of the

CSP is to find a χ-assignment x_V such that

$$\prod_{c \in C} [x_{V:V(c)} \in \Gamma_c] = 1, \tag{3.1}$$

or to conclude that no such assignment exists.

Now it is easy to verify that the factorization structure of (3.1) can be represented by a factor graph [21]: in the factor graph, "variable vertices" are indexed by V, where the "variable" indexed by $v \in V$ represents an elementary assignment $x_{V:\{v\}}$ on $\{v\}$, or simply x_v; "function vertices" are indexed by C, where the function indexed by $c \in C$ is $[x_{V:V(c)} \in \Gamma_c]$, which, with a slight overloading of notation, will also be denoted by $\Gamma_c(x_{V(c)})$; there is an edge connecting variable vertex x_v with function vertex Γ_c if and only if $v \in V(c)$. Inspired by its correspondence (to an edge) in the factor graph, we will use $(v-c)$ to denote a coordinate-constraint pair (v, c) where coordinate v is involved in constraint Γ_c in the CSP.

For notational symmetry, we denote the set $\{c : v \in V(c)\}$ by $C(v)$, namely, $C(v)$ indexes the set of all constraints involving coordinate v, or the set of all function vertices connecting to variable vertex x_v. We will assume that $|C(v)| \geq 2$ for all $v \in V$. It is clear that such an assumption is without loss of generality, since if a variable x_v is involved in only one constraint, one may always modify the constraint and remove the variable from the problem. Similarly, we will assume that $|V(c)| \geq 2$ for every $c \in C$. This is also without loss of generality since if a constraint Γ_c only involves a single variable x_v, it is always possible to "absorb" this constraint in other constraints involving x_v (noting that x_v must have another constraint since $|C(v)| \geq 2|$).

3.1.1 k-SAT

The k-SAT problems are a classic family of CSPs, known to be NP-complete for $k \geq 3$ [13]. An instance of k-SAT problems consists of a set of variables $\{x_v : v \in V\}$, each of which takes on values from the set $\chi := \{0, 1\}$, and a set of constraints $\{\Gamma_c : c \in C\}$, each of which involves

exactly k variables. For each constraint Γ_c and every $v \in V(c)$, there is a value $L_{v,c} \in \{0,1\}$ which we will refer to as the *preferred value* on v in constraint Γ_c. The k-SAT problem is then to decide on an assignment x_V such that for each constraint Γ_c, at least one of its involved coordinate is assigned its preferred value in Γ_c. To map back to the afore-mentioned set-theoretic formulation of constraints, in a k-SAT problem, for each $c \in C$, let l^c denote the χ-assignment on $V(c)$ in which every coordinate $v \in V(c)$ is assigned the negated value $\bar{L}_{v,c}$ of its preferred value $L_{v,c}$ in Γ_c, namely that $l^c_{:\{v\}} = \bar{L}_{v,c}$ for every $(v-c)$, then constraint Γ_c is defined as $\Gamma_c := \chi^{V(c)} \setminus \{l^c\}$.

For k-SAT problems, it is convenient to treat each preferred value $L_{v,c}$ as the label for edge (x_v, Γ_c) on the factor graph, and use dashed edge to represent label 0 and solid edge to represent label 1.

Example 5 *Figure 3.1 shows the factor-graph representation of a toy* 3*-SAT problem specified by* $(x_1 \vee \overline{x}_2 \vee \overline{x}_4) \wedge (x_1 \vee x_3 \vee \overline{x}_5) \wedge (x_2 \vee x_4 \vee x_5)$*. Logic operation notations are used here to define the problem, where* $\vee$ *denotes logic* `OR`*,* $\wedge$ *denotes logic* `AND`*, and the horizontal bar on a variable denotes the negation of the variable. The function represented by the factor graph is* $[(x_1, x_2, x_4) \in \Gamma_a] \cdot [(x_1, x_3, x_5) \in \Gamma_b] \cdot [(x_2, x_4, x_5) \in \Gamma_c]$*, where* $\Gamma_a = \chi^{\{1,2,4\}} \setminus \{(0_1, 1_2, 1_4)\}$*,* $\Gamma_b = \chi^{\{1,3,5\}} \setminus \{(0_1, 0_3, 1_5)\}$*, and* $\Gamma_c = \chi^{\{2,4,5\}} \setminus \{(0_2, 0_4, 0_5)\}$*.*

We note that it is customary in this thesis that variable vertices in a factor graph are listed on the left side and function (constraint) vertices listed on the right side.

3.1.2 Graph Coloring

Graph coloring or q-COL problems are another family of NP-complete problems. Given an undirected graph (Δ, Ξ) with vertex set Δ and edge set Ξ, the objective of the q-COL problem on (Δ, Ξ) is to assign each vertex in Δ a color from q different colors such that every pair of adjacent vertices have different colors. To use the above generic formulation of CSPs, we will denote the set of all q colors by set $\chi := \{1, 2, \ldots, q\}$. We will denote every undirected edge in

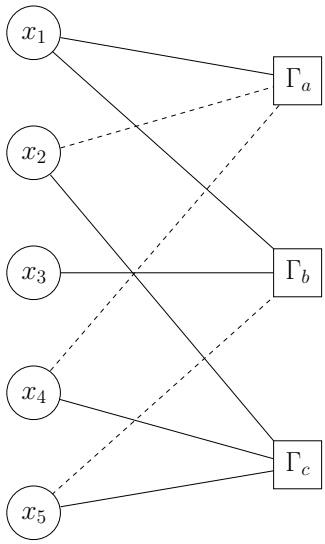


Figure 3.1: A factor graph representing a 3-SAT problem in Example 5.

Ξ, say the edge connecting vertices u and v, by set $\{u, v\}$. The set V of all coordinates is then identified with set Δ, and the set C indexing all constraints is identified with Ξ. Specifically note that every $c \in C$ is then identified with some $\{u, v\} \in \Xi$, and $V(c)$ is identified with c, or the corresponding set $\{u, v\}$. Suppose that $c = \{u, v\} \in \Xi$, then constraint Γ_c is identified with $\chi^{\{u,v\}} \setminus \{(1_u, 1_v), (2_u, 2_v), \ldots, (q_u, q_v)\}$.

Example 6 *Figure 3.2(b) shows the factor-graph representation of a q-COL problem on the undirected graph shown in Figure 3.2(a). The global function represented by the factor graph is* $[(x_1, x_2) \in \Gamma_{\{1,2\}}] \cdot [(x_1, x_3) \in \Gamma_{\{1,3\}}] \cdot [(x_2, x_3) \in \Gamma_{\{2,3\}}] \cdot [(x_3, x_4) \in \Gamma_{\{3,4\}}]$, *where* $\Gamma_{\{u,v\}} := \chi^{\{u,v\}} \setminus \{(1_u, 1_v), (2_u, 2_v), \ldots, (q_u, q_v)\}$.

3.2 Survey Propagation Algorithms

3.2.1 Survey Propagation for k-SAT Problems

Extensive study has been carried out to understand the hardness of k-SAT problems (for $k \geq 3$) and to develop efficient solvers. A parameter $\theta := |C|/|V|$ is observed to be critically related

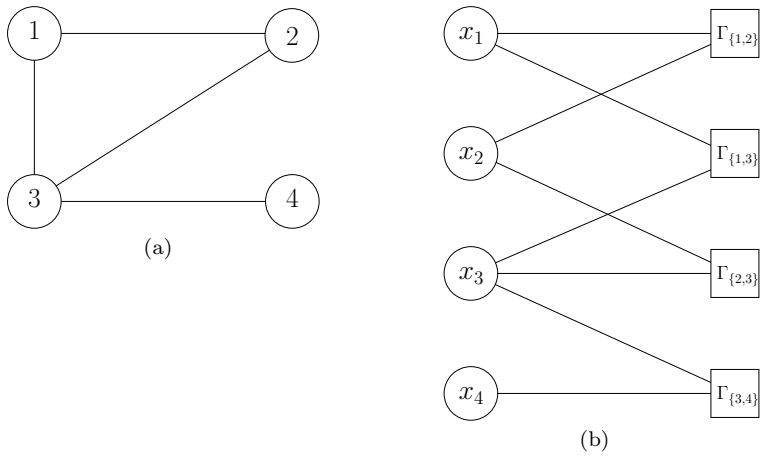


Figure 3.2: (a) An undirected graph. (b) The factor graph for a q-COL problem on graph (a).

to the hardness of random k-SAT problems. There appear two thresholds of θ, denoted by θ_d and θ_c, $(\theta_d < \theta_c)$, marking two "phase transitions" [28]. When $\theta > \theta_c$, random k-SAT problems are unsatisfiable (i.e., having no satisfying assignment) with high probability; when $\theta_d < \theta < \theta_c$, the satisfying assignments form exponentially many disjoint "clusters", making the problem extremely difficult; when $\theta < \theta_d$, the satisfying assignments merge into one huge cluster and problems are easier. In the regime of $\theta < \theta_d$, local search algorithms, such as BP, may find a satisfying assignment. In the regime of $\theta_d < \theta < \theta_c$, local search algorithms usually fail. We remark that in addition to what is noted above, a large body of literature exists discussing the structure of the solution space of random CSPs, and the reader is referred to [1–3, 5, 12, 14, 15, 18, 20, 26, 27, 30, 31, 43] for example.

The discovery and first application of survey propagation (SP) are in solving the k-SAT problems in the hard regime, where messages are passed on the above-defined factor graphs [28]. In SP, a "joker" symbol "$*$" is introduced to variable alphabet χ of the k-SAT problem, where x_v equal to the "joker" indicates that it is free to take any value from its original alphabet, and that x_v equals a non-joker symbol indicates that it is constrained to taking the designated value. Briefly, SP on k-SAT problems may be viewed as an iterative method for estimating

the "biases" of each variable x_v on $0, 1$ and $*$ respectively and a variable that is highly biased on 0 or 1 can be fixed to that value whereby simplifying the problem. It is shown that in the hard regime of random k-SAT problems, the "joker" symbol connects the disconnected clusters, making SP remain very effective even for θ very close to θ_c [23]. For k-SAT problems, the original version of SP [28] is generalized in [23] to what we call the *weighted SP* [1] or SP(ϵ) in this thesis. SP(ϵ) is a family of algorithms parametrized by a real number $\epsilon \in [0, 1]$, where SP(1) is the original SP and for some judicious choice of $\epsilon \in (0, 1)$, SP(ϵ) may have further improved performance.

We note that generalizing SP to the family of weighted SP algorithms has only been reported for k-SAT problems to date, and one of the objectives of this research is to extend such a generalization to arbitrary CSPs.

Similar to BP, in the SP algorithms, messages are passed between variable vertices and function vertices. For the purpose of describing the SP message-update rule for k-SAT problems, we introduce the following notations. For any $(v - c)$, $C_c^{\mathtt{u}}(v)$ denotes the set $\{b \in C(v) \setminus \{c\} : L_{v,b} \neq L_{v,c}\}$, and $C_c^{\mathtt{s}}(v)$ denotes the set $\{b \in C(v) \setminus \{c\} : L_{v,b} = L_{v,c}\}$.

Following [23], the message-update rule of SP(ϵ) is described as follows.

The message passed from variable vertex x_v to function vertex Γ_c — also referred as a *left message* — is a triplet of real numbers $(\Pi^{\mathtt{u}}_{v\to c}, \Pi^{\mathtt{s}}_{v\to c}, \Pi^{*}_{v\to c})$, and the message passed from function vertex Γ_c to variable vertex x_v — also referred to as a *right message* — is a real number $\eta_{c\to v} \in [0, 1]$. These messages are updated respectively according to the following equations.

[1] In [23], weighted SP is referred to as generalized SP. In this thesis, we would like to reserve the term "generalized SP" to refer to SP algorithms generalized for arbitrary CSPs beyond k-SAT problems.

$$\Pi^{\mathtt{u}}_{v\to c} := \left(1-\epsilon\prod_{b\in C^{\mathtt{u}}_c(v)}(1-\eta_{b\to v})\right)\prod_{b\in C^{\mathtt{s}}_c(v)}(1-\eta_{b\to v}) \tag{3.2}$$

$$\Pi^{\mathtt{s}}_{v\to c} := \left(1-\prod_{b\in C^{\mathtt{s}}_c(v)}(1-\eta_{b\to v})\right)\prod_{b\in C^{\mathtt{u}}_c(v)}(1-\eta_{b\to v}) \tag{3.3}$$

$$\Pi^{*}_{v\to c} := \prod_{b\in C^{\mathtt{s}}_c(v)}(1-\eta_{b\to v})\prod_{b\in C^{\mathtt{u}}_c(v)}(1-\eta_{b\to v}) \tag{3.4}$$

$$\eta_{c\to v} := \prod_{u\in V(c)\setminus\{v\}}\frac{\Pi^{\mathtt{u}}_{u\to c}}{\Pi^{\mathtt{u}}_{u\to c}+\Pi^{\mathtt{s}}_{u\to c}+\Pi^{*}_{u\to c}}. \tag{3.5}$$

The initialization of SP messages is usually random, and message-passing schedule is typically similar to the *flooding schedule* [21] in BP message passing, namely, that each iteration may be defined by all variable vertices passing messages followed by all function vertices passing messages. We note that throughout this thesis all message-passing schedules are restricted to the flooding schedule for convenience, where each iteration is defined as first updating all "left messages" and then updating all "right messages" [2].

Similar to BP, at the end of an iteration, SP may compute a "summary message" at each variable vertex. For any $v \in V$, define $C^1(v) := \{b \in C(v) : L_{v,b} = 1\}$ and $C^0(v) := \{b \in C(v) : L_{v,b} = 0\}$, then the "summary message" at x_v is a triplet $(\zeta^1_v, \zeta^0_v, \zeta^*_v)$ of real numbers, computed by

$$\zeta^1_v := \left(1-\epsilon\prod_{b\in C^1(v)}(1-\eta_{b\to v})\right)\prod_{b\in C^0(v)}(1-\eta_{b\to v}) \tag{3.6}$$

$$\zeta^0_v := \left(1-\epsilon\prod_{b\in C^0(v)}(1-\eta_{b\to v})\right)\prod_{b\in C^1(v)}(1-\eta_{b\to v}) \tag{3.7}$$

$$\zeta^*_v := \epsilon\prod_{b\in C^1(v)}(1-\eta_{b\to v})\prod_{b\in C^0(v)}(1-\eta_{b\to v}) \tag{3.8}$$

[2] An iteration may also include updating all summary messages after updating the right messages; see the description of summary messages.

where summary message $(\zeta^1, \zeta^0, \zeta^*)$ is typically normalized to a scaled version $(\zeta^{1\text{norm}}, \zeta^{0\text{norm}}, \zeta^{*\text{norm}})$ such that

$$\zeta^{1\text{norm}} + \zeta^{0\text{norm}} + \zeta^{*\text{norm}} = 1.$$

Equations (3.2) to (3.8) and the normalization procedure after completely specify the message-update rule of SP(ϵ).

Usually, SP is applied in conjunction with a heuristic "decimation" procedure, which is carried out after SP converges or after a certain number of SP iterations. In the decimation procedure, the "polarity" $B(v) := \zeta_v^{0\text{norm}} - \zeta_v^{1\text{norm}}$ at each $v \in V$ is calculated, and the most polarized variable (namely, one having the highest $|B(v)|$) is fixed to 0 or 1 according to the sign of $B(v)$: x_v is set to 0 if $B(v) > 0$, and to 1 otherwise. The k-SAT problem is then simplified and SP is applied again. This process iterates until the reduced problem is simple enough for a local search algorithm.

When $\epsilon = 1$, it is shown in [8] and [23] that the passed messages as in (3.2) through (3.5) can be interpreted probabilistically, namely, $\eta_{c\to v}$ may be interpreted as the probability that a "warning" symbol is sent from Γ_c to x_v, and $\Pi^{\mathtt{u}}_{v\to c}$, $\Pi^{\mathtt{s}}_{v\to c}$ and $\Pi^{*}_{v\to c}$ are respectively the probabilities that x_v sends to Γ_c symbol $\bar{L}_{v,c}$, symbol $L_{v,c}$ and symbol $*$.

When $\epsilon < 1$, SP(ϵ) however can no longer be interpreted probabilistically. We now present a slightly modified formulation of SP(ϵ), referred to as SP*(ϵ), which is completely equivalent to SP(ϵ) defined in [23], and which will be shown in a later chapter to have a natural probabilistic interpretation.

In SP*(ϵ), the left message $(\Pi^{\mathtt{u}}_{v\to c}, \Pi^{\mathtt{s}}_{v\to c}, \Pi^{*}_{v\to c})$ passed from variable vertex x_v to function vertex Γ_c is modified to the equations given in (3.9) to (3.11), and the right message $\eta_{c\to v}$ passed from function vertex Γ_c to variable vertex x_v and the summary message $(\zeta_v^1, \zeta_v^0, \zeta_v^*)$ at variable x_v stay unchanged.

$$\Pi^{\mathtt{u}}_{v\to c} := \left(1-\epsilon\prod_{b\in C^{\mathtt{u}}_c(v)}(1-\eta_{b\to v})\right)\prod_{b\in C^{\mathtt{s}}_c(v)}(1-\eta_{b\to v}) \tag{3.9}$$

$$\Pi^{\mathtt{s}}_{v\to c} := \left(1-\epsilon\prod_{b\in C^{\mathtt{s}}_c(v)}(1-\eta_{b\to v})\right)\prod_{b\in C^{\mathtt{u}}_c(v)}(1-\eta_{b\to v}) \tag{3.10}$$

$$\Pi^{*}_{v\to c} := \epsilon\prod_{b\in C^{\mathtt{s}}_c(v)}(1-\eta_{b\to v})\prod_{b\in C^{\mathtt{u}}_c(v)}(1-\eta_{b\to v}) \tag{3.11}$$

The following lemma shows that SP(ϵ) and SP*(ϵ) are equivalent.

Lemma 1 *For the same initialization of* $\{\eta_{c\to v}:\forall(v-c)\}$*, at any given iteration, SP*$^*(\epsilon)$ *and SP*(ϵ) *give rise to identical results in* $\eta_{c\to v}$ *for every* $(v-c)$*, and in* $(\zeta^1_v,\zeta^0_v,\zeta^*_v)$ *for every* $v\in V$*.*

Proof: The lemma follows from the fact that in the computation of $\eta_{c\to v}$ and hence of $(\zeta^1_v,\zeta^0_v,\zeta^*_v)$, $\Pi^{\mathtt{s}}_{v\to c}$ and $\Pi^*_{v\to c}$ always appear together in the form of $\Pi^{\mathtt{s}}_{v\to c}+\Pi^*_{v\to c}$. But it is easy to see that in SP(ϵ) and in SP*(ϵ), $\Pi^{\mathtt{s}}_{v\to c}+\Pi^*_{v\to c}$ has the same parametric form, both equal to $\prod_{b\in C^{\mathtt{u}}_c(v)}(1-\eta_{b\to v})$. ■

We conclude this subsection by remarking that it is possible to verify that all results concerning SP(ϵ) in [23] hold for SP*(ϵ). As such, in the rest of this thesis, SP*(ϵ) rather than SP(ϵ) will be taken as the weighted SP for k-SAT problems.

3.2.2 Survey Propagation for q-COL Problems

Similar to SP developed for k-SAT problems, in q-COL problems, SP passes messages between the variable vertices and the function (constraint) vertices in the factor-graph representation of the problem. Some notable differences however exist.

First, weighted SP has not been developed for q-COL problems to date, and it is not even clear whether such algorithm family, if existing, can be developed in a similar manner as that for k-SAT in [23], namely, via reducing the BP algorithm derived from a properly defined MRF. Answering this question in a later section, we here therefore only review the original version of

SP applied to 3-COL problems following the formulation in [10], which is analogous to SP(1), or the non-weighted SP, in the context of k-SAT.

Second, the SP messages for q-COL problems can be expressed more compactly, due to a specific nature of the problem, on which we now elaborate.

For q-COL problems, each constraint vertex has degree 2. This allows the combination of the message passed from variable x_u to a neighboring constraint, say Γ_c, with the message passed from constraint Γ_c to the other neighbor, say x_v, of Γ_c. As a consequence, Γ_c may be suppressed in the factor graph, and messages are directly passed between variable vertices that are distance 2 apart [3] (or equivalently, messages are passed on graph (Δ, Ξ)). Following [10], a compact version of SP message-passing rule for 3-COL problems is given as follows, where the message passed from variable x_u to variable x_v is a quadruplet of real numbers $(\eta^1_{u\to v}, \eta^2_{u\to v}, \eta^3_{u\to v}, \eta^*_{u\to v})$. For $i = 1, 2, 3$,

$$\eta^i_{u\to v} := \frac{\prod\limits_{w\in N(u)\backslash\{v\}}(1-\eta^i_{w\to u}) - \sum\limits_{j\neq i}\prod\limits_{w\in N(u)\backslash\{v\}}(\eta^*_{w\to u}+\eta^j_{w\to u}) + \prod\limits_{w\in N(u)\backslash\{v\}}\eta^*_{w\to u}}{\sum\limits_{j=1,2,3}\prod\limits_{w\in N(u)\backslash\{v\}}(1-\eta^j_{w\to u}) - \sum\limits_{j=1,2,3}\prod\limits_{w\in N(u)\backslash\{v\}}(\eta^*_{w\to u}+\eta^j_{w\to u}) + \prod\limits_{w\in N(u)\backslash\{v\}}\eta^*_{w\to u}} \tag{3.12}$$

where $N(u)$ is the set $\{v : v \in V, \{u, v\} \in \Xi\}$, namely, the set of neighboring vertices of vertex u on graph $\{\Delta, \Xi\}$; and

$$\eta^*_{u\to v} := 1 - \sum_{j=1,2,3}\eta^j_{u\to v}. \tag{3.13}$$

For 3-COL problems, the "summary message" computed at each variable vertex x_v is a quadruplet of real numbers, denoted by $(\zeta^1_v, \zeta^2_v, \zeta^3_v, \zeta^*_v)$, where for $i = 1, 2, 3$,

$$\zeta^i_v := \frac{\prod\limits_{u\in N(v)}(1-\eta^i_{u\to v}) - \sum\limits_{j\neq i}\prod\limits_{u\in N(v)}(\eta^*_{u\to v}+\eta^j_{u\to v}) + \prod\limits_{u\in N(v)}\eta^*_{u\to v}}{\sum\limits_{j=1,2,3}\prod\limits_{u\in N(v)}(1-\eta^j_{u\to v}) - \sum\limits_{j=1,2,3}\prod\limits_{u\in N(v)}(\eta^*_{u\to v}+\eta^j_{u\to v}) + \prod\limits_{u\in N(v)}\eta^*_{u\to v}}$$

[3] Still implementing the flooding schedule, the SP message-update rule for 3-COL problems however suppresses the passing of one set of messages (say, for example, the right messages) by including the computation of these messages in updating the other set of messages.

and

$$\zeta_v^* := 1 - \sum_{j=1,2,3} \zeta_v^j.$$

Similar to that for k-SAT problems, the summary message for a 3-COL problem at variable x_v may indicate the "bias" of variable x_v to each letter in $\{1, 2, 3, *\}$. In the decimation procedure for 3-COL problems – carried out in a similar way to that for k-SAT problems, a variable is fixed to a color $i \in \{1, 2, 3\}$ if it is highly biased to that color. The reader is referred to [10] for a detailed account of a heuristic decimation rule used in solving 3-COL problems using SP.

We note that this research primarily focuses on SP update equations, where the decimation aspect of SP is less emphasized. Later in Chapter 6 when we devise an algorithm for coloring problems, some of this aspect will be considered.

3.3 Concluding Remarks

This chapter primarily consists of a review of CSPs and SP algorithms. Using the examples of k-SAT problems and q-COL problems, we illustrate the algorithmic procedures of non-weighted SP and weighted SP and their power in solving such prototypical NP-complete problems.

A modest contribution of this chapter may include

- a generic formulation of CSPs on factor graphs, and
- a reformulation of the weighted SP algorithm presented in [23].

We note that the latter aspect above, never reported in the literature prior to this work, is in fact an important step towards generalizing SP — the main theme of this thesis.

Chapter 4

Generalized Survey Propagation

To date, SP algorithms have been applied to various other CSPs, for example, in coding for Blackwell channels [42], in quantization of Bernoulli sources [40], and in solving graph coloring problems [10], etc.. However, a general formulation of SP, particularly that of weighted SP, for solving arbitrary non-binary CSPs, has been largely missing. Specifically, we note the following milestones in the formulation of SP algorithms.

- The work of [8] presents non-weighted version of SP formulas for general CSPs beyond those involving only binary variables. However, the exposition of [8] uses the language of statistical physics, rather remote to the engineering community, and a cleaner and more friendly formulation of SP, and particularly of weighted SP, is desirable for general problems.

- The work of [23] presents weighted SP for k-SAT problems, in which weighted SP is treated as a special case of BP in a properly defined MRF. This treatment of SP and the corresponding principle for developing weighted SP are conceivably applicable to all binary CSPs. However, it has remained open, prior to our research, whether such an approach to understanding and developing weighted SP is applicable to arbitrary non-binary CSPs.

The line of development in this chapter is summarized below.

We will first present an understanding of non-weighted SP for arbitrary CSPs (namely, that formulated in [8]) in terms of "probabilistic token passing" (PTP). Although similar understanding has been previously reported in various contexts, we here stress the role of extending the variable alphabet in SP algorithms, and explicitly point out that the alphabet extension is *not* to simply include an extra joker symbol, but to *replace* the variable alphabet with its *power set* (excluding the empty-set element). To make the PTP procedure more intuitively sensible, prior to defining PTP, we will introduce a precursor of PTP, which we call "deterministic token passing" (DTP).

After introducing PTP, we then show that the probabilistic interpretation of non-weighted SP in terms of PTP makes it naturally generalizable to a weighted version, which we call weighted PTP. For a brief preview, the generalization of PTP to weighted PTP essentially involves generalizing a *functional dependency* in PTP message-update rule to a *probabilistic dependency*. Interestingly as we will show, it turns out that for k-SAT problems, weighted PTP precisely coincides with weighted SP of [23]. This should convincingly demonstrate that weighted PTP is a generalization of weighted SP for arbitrary CSPs.

A preliminary investigation of the dynamics of PTP and weighted PTP is carried out and some results, which may provide intuitions on how weighted PTP works, are also reported.

The outline of this chapter is given as follows. Section 4.1 introduces the notion of alphabet extension and related concepts. Section 4.2 defines DTP as a precursor of PTP and relate it to the max-product algorithm. In Section 4.3, we introduce PTP and show that PTP is equivalent to SP, using 3-COL problem as an example. In Section 4.4, we introduce weighted PTP and show that weighted PTP generalize weighted SP using k-SAT problems as an example. In Section 4.5, we present results pertaining to the dynamics of PTP and weighted PTP. We briefly conclude the chapter in Section 4.6.

4.1 Alphabet Extension

For a given CSP with variable alphabet χ, we define the *extended alphabet* χ^* as the power set of χ excluding the empty set $\emptyset$. That is, $\chi^* = \{t : t \subseteq \chi, t \neq \emptyset\})$. The extended alphabet χ^* of k-SAT problems is then the set $\{\{0\}, \{1\}, \{0, 1\}\}$. For 3-COL problems, χ^* is the set $\{\{1\}, \{2\}, \{3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}, \{1, 2, 3\}\}$. Each element t of χ^* will be written as a string – in bold font – containing the elements of t. For example, we may write $\{1, 2\}$ as **12**, $\{1, 2, 3\}$ as **123** and $\{1\}$ simply as **1**.

Given any subset $U \subseteq V$, a χ^*-assignment y_U on U is referred to as a *rectangle* on U. The set of all rectangles on U is denoted by $(\chi^*)^U$. Given rectangle $y_U \in (\chi^*)^U$, for every $v \in U$, $y_{U:\{v\}}$, or simply written as y_v — following an earlier convention of this thesis — is referred to as the v-*side* of y_U. Apparently, rectangle y_U has $|U|$ sides, and may also be written as the concatenation of all its sides, namely, as $\langle y_v \rangle_{v \in U}$.

For any $v \in V$, an elementary χ^*-assignment $t_v \in (\chi^*)^{\{v\}}$ will be referred to as a *token* on v. Using this nomenclature, the v-side of any rectangle is a token on v. We note that a token t_v may be interpreted as a set of elementary χ-assignments on $\{v\}$, which is in fact the set of all elementary χ-assignments on $\{v\}$ that assign v a value in set $t_v(v) \subseteq \chi$. For example, suppose that $\chi := \{1, 2, 3\}$, then token $\mathbf{12}_v$ may be identified with the set $\{1_v, 2_v\}$ of elementary χ-assignments on $\{v\}$.

It is worth noting that when a token t_v is identified with a set of elementary χ-assignments on v, a rectangle $\langle t_v \rangle_{v \in U}$ may be identified with the *Cartesian product* of all its sides. For example, rectangle $(\mathbf{12}_v, \mathbf{23}_u)$ may be interpreted as the following set of χ-assignments on $\{v, u\}$: $\{(1_v, 2_u), (1_v, 3_u), (2_v, 2_u), (2_v, 3_u)\}$. Under this interpretation, we will also make frequent uses of the Cartesian product notation, writing rectangle $(\mathbf{12}_v, \mathbf{23}_u)$ as $\mathbf{12}_v \times \mathbf{23}_u$, and rectangle $\langle t_v \rangle_{v \in U}$ as $\prod_{v \in U} t_v$. We note that this interpretation is in fact the reason for which we choose the terminologies "rectangle" and "side".

For simplicity, from here on, we shall reserve the term "assignment" for referring to a χ-assignment only, and a χ^*-assignment will be referred to as a "rectangle", "side" or "token".

We say that an assignment x_U on U is *contained* in rectangle y_U if $x_{U:\{v\}}(v) \in y_{U:\{v\}}(v)$ for every $v \in U$. For example, assignment $(1_v, 2_u)$ is contained in rectangle $(\mathbf{13}_v, \mathbf{23}_u)$ We will use $x_U \in y_U$ to denote this containedness relationship, since this notation is precise when the rectangle y_U is interpreted as a *set* of assignments on U.

Given a CSP and a $(v - c)$ pair, we define function $\mathtt{F}_c^v : (\chi^*)^{V(c)\setminus\{v\}} \to (\chi^*)^{\{v\}}$ as follows: for every rectangle $\prod_{u\in V(c)\setminus\{v\}} t_u$ on $V(c) \setminus \{v\}$,

$$\mathtt{F}_c^v \left(\prod_{u\in V(c)\setminus\{v\}} t_u \right) := \left(\left(\chi^{\{v\}} \times \prod_{u\in V(c)\setminus\{v\}} t_u \right) \cap \Gamma_c \right)_{:\{v\}}.$$

We often write $\mathtt{F}_c^v$ in short as $\mathtt{F}_c$ since the domain and co-domain of the function may be recovered from the form of its argument. Given rectangle $\prod_{u\in V(c)\setminus\{v\}} t_u$ on $V(c) \setminus \{v\}$, we call $\mathtt{F}_c \left(\prod_{u\in V(c)\setminus\{v\}} t_u \right)$ the *forced token* by rectangle $\prod_{u\in V(c)\setminus\{v\}} t_u$ via constraint Γ_c. It is easy to verify that the forced token $\mathtt{F}_c \left(\prod_{u\in V(c)\setminus\{v\}} t_u \right)$ is simply the set of all (elementary) assignments on $\{v\}$ which, when concatenated with an assignment on $V(c) \setminus \{v\}$ contained in rectangle $\prod_{u\in V(c)\setminus\{v\}} t_u$, make local constraint Γ_c satisfied. We now give some examples using the toy 3-SAT problem shown in Figure 3.1 to illustrate this definition. Consider constraint Γ_a, if rectangle $t_{\{1,2\}}$ on $\{1, 2\}$ is defined as $(\mathbf{1}_1, \mathbf{01}_2)$, then forced token $\mathtt{F}_a(t_{\{1,2\}}) = \mathbf{01}_4$, since when assigning variable x_4 either value 0 or 1, it is possible to find an assignment of variables x_1 and x_2 in rectangle $t_{\{1,2\}}$ that makes Γ_a satisfied; on the other hand, if $t_{\{1,2\}} = (\mathbf{0}_1, \mathbf{1}_2)$, then forced token $\mathtt{F}_a(t_{\{1,2\}}) = \mathbf{0}_4$, since rectangle $t_{\{1,2\}}$ contains a single assignment of x_1 and x_2 (namely $(0_1, 1_2)$), and the only assignment of x_4 that will make constraint Γ_a satisfied is the one assigning 0 to x_4, namely 0_4.

A "monotonicity property" of function $\mathtt{F}_c$, stated in the following lemma, follows immediately from the definition of the function.

Lemma 2 *Suppose that x_v and Γ_c are a pair of neighboring variable and constraint vertices in the factor graph, and that $y_{V(c)\setminus\{v\}}$ and $y'_{V(c)\setminus\{v\}}$ are two rectangles on $V(c) \setminus \{v\}$. Then*

$y_{V(c)\backslash\{v\}} \subset y'_{V(c)\backslash\{v\}}$ *implies that* $\mathtt{F}_c\left(y_{V(c)\backslash\{v\}}\right) \subseteq \mathtt{F}_c\left(y'_{V(c)\backslash\{v\}}\right)$.

Proof: The proof of this lemma is simply based on the definition of forced token.

$$\begin{aligned} & y_{V(c)\backslash\{v\}} \subset y'_{V(c)\backslash\{v\}} \\ \Rightarrow \quad & \chi^{\{v\}} \times y_{V(c)\backslash\{v\}} \subset \chi^{\{v\}} \times y'_{V(c)\backslash\{v\}} \\ \Rightarrow \quad & \left(\chi^{\{v\}} \times y_{V(c)\backslash\{v\}}\right) \cap \Gamma_c \subseteq \left(\chi^{\{v\}} \times y'_{V(c)\backslash\{v\}}\right) \cap \Gamma_c \\ \Rightarrow \quad & \mathtt{F}_c\left(y_{V(c)\backslash\{v\}}\right) \subseteq \mathtt{F}_c\left(y'_{V(c)\backslash\{v\}}\right). \end{aligned}$$

■

4.2 Deterministic Token Passing (DTP)

As we will introduce — for arbitrary CSPs — a probabilistic interpretation of non-weighted SP (namely, PTP) and generalize it to a weighted version (namely, weighted PTP), in this section, we first introduce an algorithmic procedure, which we call *deterministic token passing* or DTP. DTP is also referred to as "warning propagation" in statistical physics literature of SP algorithms [9]. We note that the purpose of introducing DTP is to provide an easier access to PTP, a procedure to be introduced in the next section.

In DTP, messages are tokens passed along the edges of the factor graph representing the CSP of interest. Specifically, the token passed from and to each variable x_v is a token on v, or equivalently, a set of (elementary) assignments on $\{v\}$. For any pair of neighboring vertices x_v and Γ_c on the factor graph, the token, or left message, $t_{v\to c}$ passed from variable x_v to constraint Γ_c depends on all incoming tokens (right messages) passed to x_v except the one passed from Γ_c. Similarly, the token, or right message, $t_{c\to v}$ passed from constraint Γ_c to variable x_v depends on all incoming tokens (left messages) passed to Γ_c except the one passed from x_v. Each iteration of token passing in DTP is defined by every variable passing a token on each of its edges followed by every constraint passing a token on each of its edges. Within

any iteration, the token-passing rule of DTP is given as follows.

$$t_{v\to c} := \bigcap_{b\in C(v)\setminus\{c\}} t_{b\to v} \tag{4.1}$$

$$t_{c\to v} := \mathtt{F}_c\left(\prod_{u\in V(c)\setminus\{v\}} t_{u\to c}\right). \tag{4.2}$$

That is, the token passed from a variable is the *intersection* of its incoming tokens from the upstream, whereas the token passed from a constraint is the forced token via the constraint by the rectangle formed by the upstream incoming tokens as sides.

It is intuitive to illuminate this message-passing rule using the following analogy. We may view the token sent from a variable as the "intention" of the variable, indicating the possible values that the variable intends to take. On the other hand, we may view the token sent from a constraint as the "command" from the constraint, indicating the possible values that the constraint allows the destination variable to take. If a is an intention and b is a command, where both are tokens on the same coordinate, then the relationship $a \subseteq b$ may be viewed as that "intention a obeys command b". Under this perspective, the token sent from a variable is the "maximal" intention of the variable that obeys all incoming commands from the upstream constraints; on the other hand, the token sent from a constraint is the "maximal" command that is "compatible" with all incoming intentions from the upstream variables. Here "maximality" is in the sense of maximizing the cardinality of the subset of assignments, and "compatibility" is in the sense of satisfying the local constraint.

Examples of token passing for a 3-COL problem are illustrated in Figure 4.1.

A summary message or "summary token" at variable vertex x_v may be computed, according to the rule in (4.3) for each $v \in V$ at any iteration after the all constraint vertices have passed tokens.

$$t_v := \bigcap_{b\in C(v)} t_{b\to v}. \tag{4.3}$$

Using the "intention-command" analogy, the summary token at a variable is the "maximal"

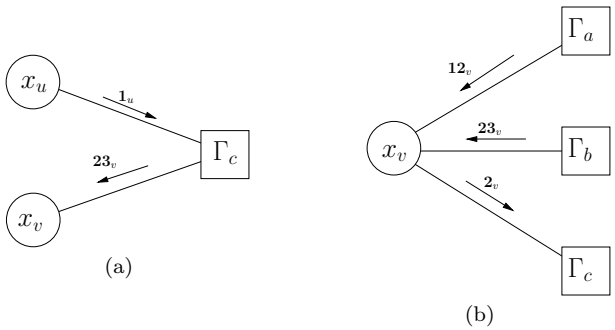


Figure 4.1: Examples of deterministic token passing for a 3-COL problem. (a) Token $t_{c\to v}$ passed from constraint Γ_c to variable x_v. (b) Token $t_{v\to c}$ passed from variable x_v to constraint Γ_c.

intention of the variable that obeys the incoming commands from *all* directions.

Some caution is needed on the well-definedness of the updating rule of passed tokens and summary tokens. That is, in (4.1), (4.2) and (4.3) the right-hand side can be equal to the empty set $\emptyset$, which is not a well-defined token. Whenever in an iteration a not-well-defined token (i.e., the empty set) arises from the updating rule, we may force DTP to terminate. — As we will see later in the "random" version of DTP (i.e., PTP and weighted PTP), we will eventually condition on the case in which these events do not happen.

At any iteration, one may read out the summary tokens at all variable vertices and form a rectangle on V using these tokens as its sides. It is clear that at any given iteration, the resulting rectangle formed by the summary tokens depends on the initialization of DTP.

4.2.1 DTP as Max-Product

To shed more light on the nature of DTP or warning propagation, we take a slight digression to show that DTP introduced above is in fact not at all a new algorithm but rather an equivalent algorithm to the well-known max-product (or min-sum) algorithm [21]. For simplicity, we will omit all proofs in this subsection.

Given the factor-graph representation of a CSP as defined in (3.1), the max-product mes-

sages passed on the factor graph are updated as follows.

$$\mu_{v\to c}^{\max-\mathrm{prod}}(x_v) := \prod_{b\in C(v)\setminus\{c\}} \mu_{v\to b}^{\max-\mathrm{prod}}(x_v) \tag{4.4}$$

$$\mu_{c\to v}^{\max-\mathrm{prod}}(x_v) := \max_{x_{V(c)\setminus\{v\}}} \left([x_{V(c)} \in \Gamma_c] \prod_{u\in V(c)\setminus\{v\}} \mu_{u\to c}^{\max-\mathrm{prod}}(x_u) \right) \tag{4.5}$$

$$\mu_{v}^{\max-\mathrm{prod}}(x_v) := \prod_{b\in C(v)} \mu_{v\to b}^{\max-\mathrm{prod}}(x_v) \tag{4.6}$$

where $\mu_{v\to c}^{\max-\mathrm{prod}}$, $\mu_{c\to v}^{\max-\mathrm{prod}}$ and $\mu_{v}^{\max-\mathrm{prod}}$, all functions on χ, are respectively a left message (i.e., message passed from variable x_v to constraint Γ_c), a right message (i.e., message passed from constraint Γ_c to variable x_v), and a summary message (i.e., message computed at variable x_v).

Lemma 3 *Max-product messages for factor graph representing (3.1) are* $\{0,1\}$*-valued functions, as long as the initial message passed from each node is a* $\{0,1\}$*-valued function.*

Proof: This lemma can be simply proved by max-product message-update rules (4.4)-(4.6). ■

We note that any $\{0,1\}$-valued function on χ is characterized by a subset, or token, $t \subseteq \chi$ at which the function equal to 1, we thus use notation Φ_t to denote such a "token indicator function", namely, $\Phi_t(x) := [x \in t]$.

Lemma 4 *1. Suppose that* $t_1, t_2, \ldots, t_n$ *are an arbitrary collection of subsets of* χ*, then*

$$\Phi_{t_1}(x)\Phi_{t_2}(x)\ldots\Phi_{t_n}(x) = \Phi_{t_1\cap t_2\cap\ldots\cap t_n}(x).$$

2. Suppose that Γ_c *is a constraint involving coordinates* $V(c)$ *and* $\{t_u : u \in V(c)\setminus\{v\}\}$ *is an arbitrary set of subsets of* χ*. Then*

$$\max_{x_{V(c)\setminus\{v\}}} \left([x_{V(c)} \in \Gamma_c] \prod_{u\in V(c)\setminus\{v\}} \Phi_{t_u}(x_u) \right) = \Phi_{\mathtt{F}_c\left(\prod_{u\in V(c)\setminus\{v\}} t_u\right)}(x_v).$$

Proof: The first part of the lemma can be immediately obtained from the definition function $\Phi_t(x)$.

$$\begin{aligned} & \Phi_{t_1}(x)\Phi_{t_2}(x)\ldots\Phi t_n(x) \\ = \;& [x \in t_1] \cdot [x \in t_2] \cdots [x \in t_n] \\ = \;& [x \in t_1 \cap t_2 \cap \ldots \cap t_n] \\ = \;& \Phi_{t_1 \cap t_2 \cap \ldots \cap t_n}(x). \end{aligned}$$

The proof of the second part of the lemma is as follow.

$$\begin{aligned} & \max_{x_{V(c)\setminus\{v\}}} \left([x_{V(c)} \in \Gamma_c] \prod_{u \in V(c)\setminus\{v\}} \Phi_{t_u}(x_u) \right) \\ = \;& \max_{x_{V(c)\setminus\{v\}}} \Phi_{\Gamma_c \cap ((\prod_{u \in V(c)\setminus\{v\}} t_u) \times \chi_v)} x_{V(c)} \\ \overset{(a)}{=} \;& \Phi_{(\Gamma_c \cap ((\prod_{u \in V(c)\setminus\{v\}} t_u) \times \chi_v)):\{v\}} x_v \\ = \;& \Phi_{\mathtt{F}_c(\prod_{u \in V(c)\setminus\{v\}} t_u)}(x_v), \end{aligned}$$

where (a) refers to the fact that given $S \subseteq \chi^W$ and $U \subset W$, $\Phi_{S:U}(x_U) = \max_{x_{W\setminus U}} \Phi_S(x_W)$.

■

Under the lemmas above, when each initial max-product message on the factor graph representing (3.1) is a token indicator function, it is easy to verify that at any iteration, the max-product messages are the indicator functions of the token passed in DTP. Such an equivalence between DTP and the max-product algorithm is precisely due to that each factor in the factor graph representation of CSPs is an indicator function. Finally we present a simple lemma relating satisfiability of a CSP to a behavior of the max-product algorithm.

Lemma 5 *Suppose that the factor graph representing (3.1) is connected and the max-product algorithm is applied on the factor graph where the initial message passed from each variable is the all-one message. If a max-product message passed on the factor graph is the all-zero*

message at some iteration, then the max-product algorithm converges (within a finite number of iterations) to the configuration where all messages are the all-zero message. When this occurs, it is necessarily the case that the CSP has no solution.

4.3 Probabilistic Token Passing (PTP)

We now introduce the "probabilistic token passing" (or PTP) procedure. The key distinction between PTP and DTP is that on each edge and along each direction, PTP passes a *random* token and the messages being updated in PTP are the *distributions* of the random tokens.

Specifically, PTP message-update rule can be constructed by considering the following mechanism of passing random tokens.

1. On each edge connecting variable x_v and constraint Γ_c in the factor graph, the token $t_{v\to c}$ passed to constraint Γ_c and the token $t_{c\to v}$ passed to variable x_v are both *random variables*, distributed over $(\chi^*)^{\{v\}}$.

2. For any given vertex in the factor graph, all of its incoming random tokens are assumed to be independent.

3. For any given vertex in the factor graph, the outgoing random token sent along any edge is a function of all the incoming random tokens from the upstream, where the functional dependency is precisely that specified in DTP, namely, (4.1) or (4.2), depending on whether the vertex is a variable vertex or a function (constraint) vertex.

4. The summary (random) token t_v at each variable vertex x_v is a function of all incoming random tokens, where the functional dependency is precisely that specified in DTP, namely, (4.3).

Building on this mechanism, we will then define each PTP (passed or summary) message as the distribution of the corresponding random token *conditioned* on that the token is well defined (namely, not equal to the empty set). We note that such a "conditioning" merely involves a

normalization (namely, scaling) of each message so that it sums to 1 over all valid tokens. We will use $\lambda_{v\to c}$ to denote the message sent from x_v to Γ_c — also referred to as a left message, $\rho_{c\to v}$ to denote the message sent from Γ_c to x_v — also referred to as a right message, and μ_v to denote the summary message at variable vertex x_v. It is then straight-forward to derive the message-update rule of PTP as follows, where the superscript "`norm`" on a message indicates that the message has been normalized.

PTP Message-Update Rule

$$\lambda_{v\to c}(t_{v\to c}) := \sum_{\langle t_{b\to v}\rangle_{b\in C(v)\setminus\{c\}}} \left[t_{v\to c} = \bigcap_{b\in C(v)\setminus\{c\}} t_{b\to v}\right] \prod_{b\in C(v)\setminus\{c\}} \rho^{\text{norm}}_{b\to v}(t_{b\to v}) \tag{4.7}$$

$$\rho_{c\to v}(t_{c\to v}) := \sum_{\langle t_{u\to c}\rangle_{u\in V(c)\setminus\{v\}}} \left[t_{c\to v} = \mathtt{F}_c\left(\prod_{u\in V(c)\setminus\{v\}} t_{u\to c}\right)\right] \prod_{u\in V(c)\setminus\{v\}} \lambda^{\text{norm}}_{u\to c}(t_{u\to c}) \tag{4.8}$$

$$\mu_v(t_v) := \sum_{\langle t_{c\to v}\rangle_{c\in C(v)}} \left[t_v = \bigcap_{c\in C(v)} t_{c\to v}\right] \prod_{c\in C(v)} \rho^{\text{norm}}_{c\to v}(t_{c\to v}), \tag{4.9}$$

and the normalized messages are defined as

$$\lambda^{\text{norm}}_{v\to c}(t_{v\to c}) := \lambda_{v\to c}(t_{v\to c}) / \sum_{t\in(\chi^*)^{\{v\}}} \lambda_{v\to c}(t) \tag{4.10}$$

$$\rho^{\text{norm}}_{c\to v}(t_{c\to v}) := \rho_{c\to v}(t_{c\to v}) / \sum_{t\in(\chi^*)^{\{v\}}} \rho_{c\to v}(t) \tag{4.11}$$

$$\mu^{\text{norm}}_{v}(t_{v}) := \mu_{v}(t_{v}) / \sum_{t\in(\chi^*)^{\{v\}}} \mu_{v}(t). \tag{4.12}$$

We note that the update of messages in each PTP iteration is proceeded by first computing the un-normalized messages and then computing their normalized version.

4.3.1 SP as PTP

We now show that SP is precisely PTP using the example of 3-COL problems. Here we note that it is possible (and entails little additional difficulty) to show the equivalence between PTP and the *general* formulation of non-weighted SP [8] for arbitrary CSPs. However, as we feel it unnecessary to introduce additional statistical physics terminologies presented in [8], we choose not to repeat the exposition of SP in [8] and only show that SP is PTP for the special case of 3-COL problems.

In the factor graph representing a 3-COL problem, noting that each constraint vertex has degree 2, we will make a slight abuse of notation: for any $(v-c)$ pair, we will use $V(c)\setminus\{v\}$ to also denote the *index* of the unique other variable vertex (besides x_v) connecting to Γ_c, although $V(c)\setminus\{v\}$ originally refers to the singleton set containing that index. Whether $V(c)\setminus\{v\}$ should be treated as the index of a variable or as the singleton set containing the index should be clear from the context.

For notational simplicity, from here on, for every element in the token set $(\chi^*)^{\{v\}}$, when no ambiguity results, we will suppress the subscript indicating the coordinate of the element. For example, we will write $\mathbf{12}_v$ as $\mathbf{12}$, when the subscript can be recovered from the context. Additionally, we will use i, j, and k to denote the three distinct colors $1, 2$, and 3 in the 3-COL problem, so that token **i** can refer to any token that is a singleton set, token **ij** can refer to any token that contains a pair of assignments, and token **ijk** refers to the token containing all three assignments.

Using these notations, the PTP message-update rule for 3-COL problems can be easily derived, which is presented in the following lemma.

Lemma 6 *For 3-COL problems, the PTP message-update rule is:*

$$\lambda_{v\to c}(\mathbf{i}) := \prod_{b\in C(v)\setminus\{c\}} (\rho^{\text{norm}}_{b\to v}(\mathbf{ij}) + \rho^{\text{norm}}_{b\to v}(\mathbf{ik}) + \rho^{\text{norm}}_{b\to v}(\mathbf{ijk})) - \prod_{b\in C(v)\setminus\{c\}} (\rho^{\text{norm}}_{b\to v}(\mathbf{ij}) + \rho^{\text{norm}}_{b\to v}(\mathbf{ijk}))$$
$$- \prod_{b\in C(v)\setminus\{c\}} (\rho^{\text{norm}}_{b\to v}(\mathbf{ik}) + \rho^{\text{norm}}_{b\to v}(\mathbf{ijk})) + \prod_{b\in C(v)\setminus\{c\}} \rho^{\text{norm}}_{b\to v}(\mathbf{ijk}) \tag{4.13}$$

$$\lambda_{v\to c}(\mathbf{ij}) := \prod_{b\in C(v)\setminus\{c\}} (\rho^{\text{norm}}_{b\to v}(\mathbf{ij}) + \rho^{\text{norm}}_{b\to v}(\mathbf{ijk})) - \prod_{b\in C(v)\setminus\{c\}} \rho^{\text{norm}}_{b\to v}(\mathbf{ijk}) \tag{4.14}$$

$$\lambda_{v\to c}(\mathbf{ijk}) := \prod_{b\in C(v)\setminus\{c\}} \rho^{\text{norm}}_{b\to v}(\mathbf{ijk}) \tag{4.15}$$

$$\rho_{c\to v}(\mathbf{ij}) := \lambda^{\text{norm}}_{V(c)\setminus\{v\}\to c}(\mathbf{k}) \tag{4.16}$$

$$\rho_{c\to v}(\mathbf{ijk}) := \lambda^{\text{norm}}_{V(c)\setminus\{v\}\to c}(\mathbf{ij}) + \lambda^{\text{norm}}_{V(c)\setminus\{v\}\to c}(\mathbf{ik}) + \lambda^{\text{norm}}_{V(c)\setminus\{v\}\to c}(\mathbf{jk}) + \lambda^{\text{norm}}_{V(c)\setminus\{v\}\to c}(\mathbf{ijk}) \tag{4.17}$$

$$\mu_v(\mathbf{i}) := \prod_{c\in C(v)} (\rho^{\text{norm}}_{c\to v}(\mathbf{ij}) + \rho^{\text{norm}}_{c\to v}(\mathbf{ik}) + \rho^{\text{norm}}_{c\to v}(\mathbf{ijk})) - \prod_{c\in C(v)} (\rho^{\text{norm}}_{c\to v}(\mathbf{ij}) + \rho^{\text{norm}}_{c\to v}(\mathbf{ijk}))$$
$$- \prod_{c\in C(v)} (\rho^{\text{norm}}_{c\to v}(\mathbf{ik}) + \rho^{\text{norm}}_{c\to v}(\mathbf{ijk})) + \prod_{c\in C(v)} \rho^{\text{norm}}_{c\to v}(\mathbf{ijk}) \tag{4.18}$$

$$\mu_v(\mathbf{ij}) := \prod_{c\in C(v)} (\rho^{\text{norm}}_{c\to v}(\mathbf{ij}) + \rho^{\text{norm}}_{c\to v}(\mathbf{ijk})) - \prod_{c\in C(v)} \rho^{\text{norm}}_{c\to v}(\mathbf{ijk}) \tag{4.19}$$

$$\mu_v(\mathbf{ijk}) := \prod_{c\in C(v)} \rho^{\text{norm}}_{c\to v}(\mathbf{ijk}). \tag{4.20}$$

It is then possible to relate the PTP messages and the (non-weighted) SP messages for 3-COL problems, and show their equivalence.

Theorem 1 *For 3-COL problems, the correspondence between SP and PTP message-update rules is*

$$
\begin{aligned}
\eta^{i}_{u\to v} &\leftrightarrow \lambda^{\text{norm}}_{u\to\{u,v\}}(\mathbf{i}) \\
\eta^{*}_{u\to v} &\leftrightarrow 1-\sum_{\mathbf{i=1,2,3}} \lambda^{\text{norm}}_{u\to\{u,v\}}(\mathbf{i}) \\
&\quad = \lambda^{\text{norm}}_{u\to\{u,v\}}(\mathbf{ij})+\lambda^{\text{norm}}_{u\to\{u,v\}}(\mathbf{ik})+\lambda^{\text{norm}}_{u\to\{u,v\}}(\mathbf{jk})+\lambda^{\text{norm}}_{u\to\{u,v\}}(\mathbf{ijk}) \\
\eta^{i}_{u} &\leftrightarrow \mu^{\text{norm}}_{u}(\mathbf{i}) \\
\eta^{*}_{u} &\leftrightarrow 1-\sum_{\mathbf{i=1,2,3}} \mu^{\text{norm}}_{u}(\mathbf{i}).
\end{aligned}
$$

Proof: First we will identify c in the subscript of $\lambda^{\text{norm}}_{u\to c}$ with $\{u,v\}$ in which v indexes the destination vertex in the subscript of $\eta_{u\to v}$.

For any $c=\{u,v\}$, let $\alpha_{u,v}=\lambda^{\text{norm}}_{u\to c}(\mathbf{ij})+\lambda^{\text{norm}}_{u\to c}(\mathbf{ik})+\lambda^{\text{norm}}_{u\to c}(\mathbf{jk})+\lambda^{\text{norm}}_{u\to c}(\mathbf{ijk})$. When applying PTP update equations (4.16) and (4.17) to equations (4.13) to (4.15) and re-writing the update rule in terms of left messages only, the un-normalized left messages are updated as follows.

$$
\begin{aligned}
\lambda_{u\to c}(\mathbf{i}) &= \prod_{b\in C(u)\backslash\{c\}}\left(1-\lambda^{\text{norm}}_{V(b)\backslash\{u\}\to b}(\mathbf{i})\right)-\prod_{b\in C(u)\backslash\{c\}}\left(\lambda^{\text{norm}}_{V(b)\backslash\{u\}\to b}(\mathbf{j})+\alpha_{V(b)\backslash\{u\},u}\right) \\
&\quad -\prod_{b\in C(u)\backslash\{c\}}\left(\lambda^{\text{norm}}_{V(b)\backslash\{u\}\to b}(\mathbf{k})+\alpha_{V(b)\backslash\{u\},u}\right)+\prod_{b\in C(u)\backslash\{c\}}\alpha_{V(b)\backslash\{u\},u} \qquad (4.21) \\
\lambda_{u\to c}(\mathbf{ij}) &= \prod_{b\in C(u)\backslash\{c\}}\left(\lambda^{\text{norm}}_{V(b)\backslash\{u\}\to b}(\mathbf{k})+\alpha_{V(b)\backslash\{u\},u}\right)-\prod_{b\in C(u)\backslash\{c\}}\alpha_{V(b)\backslash\{u\},u} \qquad (4.22) \\
\lambda_{u\to c}(\mathbf{ijk}) &= \prod_{b\in C(u)\backslash\{c\}}\alpha_{V(b)\backslash\{u\},u}. \qquad (4.23)
\end{aligned}
$$

After normalization, we have

$$\lambda^{\text{norm}}_{u\to c}(\mathbf{i}) = \frac{1}{\beta}\cdot\left(\prod_{b\in C(u)\setminus\{c\}}\left(1-\lambda^{\text{norm}}_{V(b)\setminus\{u\}\to b}(\mathbf{i})\right)-\prod_{b\in C(u)\setminus\{c\}}\left(\lambda^{\text{norm}}_{V(b)\setminus\{u\}\to b}(\mathbf{j})+\alpha_{V(b)\setminus\{u\},u}\right)\right.$$
$$\left.-\prod_{b\in C(u)\setminus\{c\}}\left(\lambda^{\text{norm}}_{V(b)\setminus\{u\}\to b}(\mathbf{k})+\alpha_{V(b)\setminus\{u\},u}\right)+\prod_{b\in C(u)\setminus\{c\}}\alpha_{V(b)\setminus\{u\},u}\right) \quad (4.24)$$

$$\lambda^{\text{norm}}_{u\to c}(\mathbf{ij}) = \frac{1}{\beta}\cdot\left(\prod_{b\in C(u)\setminus\{c\}}\left(\lambda^{\text{norm}}_{V(b)\setminus\{u\}\to b}(\mathbf{k})+\alpha_{V(b)\setminus\{u\},u}\right)-\prod_{b\in C(u)\setminus\{c\}}\alpha_{V(b)\setminus\{u\},u}\right) \quad (4.25)$$

$$\lambda^{\text{norm}}_{u\to c}(\mathbf{ijk}) = \frac{1}{\beta}\cdot\prod_{b\in C(u)\setminus\{c\}}\alpha_{V(b)\setminus\{u\},u}, \quad (4.26)$$

where $\beta := \sum_{t\in(\chi^*)^{\{u\}}}\lambda_{u\to c}(t)$.

It is easy to see that

$$\beta = \sum_{\mathbf{i}=\mathbf{1},\mathbf{2},\mathbf{3}}\prod_{b\in C(u)\setminus\{c\}}\left(1-\lambda^{\text{norm}}_{V(b)\setminus\{u\}\to b}(\mathbf{i})\right)-\sum_{\mathbf{i}=\mathbf{1},\mathbf{2},\mathbf{3}}\prod_{b\in C(u)\setminus\{c\}}\left(\lambda^{\text{norm}}_{V(b)\setminus\{u\}\to b}(\mathbf{i})+\alpha_{V(b)\setminus\{u\},u}\right)$$
$$+\prod_{b\in C(u)\setminus\{c\}}\alpha_{V(b)\setminus\{u\},u}.$$

For any $c=\{u,v\}$, it is clear that when identifying $\lambda^{\text{norm}}_{u\to c}(\mathbf{i})$ with $\eta^i_{u\to v}$ and identifying $\alpha_{\{u,v\}} = 1-\sum_{\mathbf{i}=\mathbf{1},\mathbf{2},\mathbf{3}}\lambda^{\text{norm}}_{u\to\{u,v\}}(\mathbf{i})$ with $\eta^*_{u\to v}$, the update rule for passed message $(\eta^1_{u\to v},\eta^2_{u\to v},\eta^3_{u\to v},\eta^*_{u\to v})$ in SP results.

To prove the equivalence of PTP and SP summary messages, we can follow the same procedure as we did for proving the equivalence of PTP left messages and SP left messages. When applying message update equations (4.16) and (4.17) to equations (4.18) to (4.20) and re-write summary messages in terms of left messages, the PTP summary messages are updated

as follows.

$$\begin{aligned}
\mu_u(\mathbf{i}) &= \prod_{c\in C(u)}\left(1-\lambda^{\mathrm{norm}}_{V(c)\backslash\{u\}\to c}(\mathbf{i})\right)-\prod_{c\in C(u)}\left(\lambda^{\mathrm{norm}}_{V(c)\backslash\{u\}\to c}(\mathbf{j})+\alpha_{V(c)\backslash\{u\},u}\right)\\
&\quad-\prod_{c\in C(u)}\left(\lambda^{\mathrm{norm}}_{V(c)\backslash\{u\}\to c}(\mathbf{k})+\alpha_{V(c)\backslash\{u\},u}\right)+\prod_{c\in C(u)}\alpha_{V(c)\backslash\{u\},u} \qquad (4.27)\\
\mu_u(\mathbf{ij}) &= \prod_{c\in C(u)}\left(\lambda^{\mathrm{norm}}_{V(c)\backslash\{u\}\to c}(\mathbf{k})+\alpha_{V(c)\backslash\{u\},u}\right)-\prod_{c\in C(u)}\alpha_{V(c)\backslash\{u\},u} \qquad (4.28)\\
\mu_u(\mathbf{ijk}) &= \prod_{c\in C(u)}\alpha_{V(c)\backslash\{u\},u}. \qquad (4.29)
\end{aligned}$$

After normalization, we have

$$\begin{aligned}
\mu^{\mathrm{norm}}_u(\mathbf{i}) &= \frac{1}{\beta'}\cdot\left(\prod_{c\in C(u)}\left(1-\lambda^{\mathrm{norm}}_{V(c)\backslash\{u\}\to c}(\mathbf{i})\right)-\prod_{c\in C(u)}\left(\lambda^{\mathrm{norm}}_{V(c)\backslash\{u\}\to c}(\mathbf{j})+\alpha_{V(c)\backslash\{u\},u}\right)\right.\\
&\quad\left.-\prod_{c\in C(u)}\left(\lambda^{\mathrm{norm}}_{V(c)\backslash\{u\}\to c}(\mathbf{k})+\alpha_{V(c)\backslash\{u\},u}\right)+\prod_{c\in C(u)}\alpha_{V(c)\backslash\{u\},u}\right) \qquad (4.30)\\
\mu^{\mathrm{norm}}_u(\mathbf{ij}) &= \frac{1}{\beta'}\cdot\left(\prod_{c\in C(u)}\left(\lambda^{\mathrm{norm}}_{V(c)\backslash\{u\}\to c}(\mathbf{k})+\alpha_{V(c)\backslash\{u\},u}\right)-\prod_{c\in C(u)}\alpha_{V(c)\backslash\{u\},u}\right) \qquad (4.31)\\
\mu^{\mathrm{norm}}_u(\mathbf{ijk}) &= \frac{1}{\beta'}\cdot\prod_{c\in C(u)}\alpha_{V(c)\backslash\{u\},u}, \qquad (4.32)
\end{aligned}$$

where $\beta' := \sum_{t\in(\chi^*)^{\{u\}}}\mu_u(t)$.

It is easy to show that

$$\begin{aligned}
\beta' &= \sum_{\mathbf{i}=\mathbf{1},\mathbf{2},\mathbf{3}}\prod_{c\in C(u)}\left(1-\lambda^{\mathrm{norm}}_{V(c)\backslash\{u\}\to c}(\mathbf{i})\right)-\sum_{\mathbf{i}=\mathbf{1},\mathbf{2},\mathbf{3}}\prod_{c\in C(u)}\left(\lambda^{\mathrm{norm}}_{V(c)\backslash\{u\}\to c}(\mathbf{i})+\alpha_{V(c)\backslash\{u\},u}\right)\\
&\quad+\prod_{c\in C(u)}\alpha_{V(c)\backslash\{u\},u}.
\end{aligned}$$

For any $u\in V$, it is clear that when identifying $\mu^{\mathrm{norm}}_u(\mathbf{i})$ with η^i_u and identifying $1-\sum_{\mathbf{i}=\mathbf{1},\mathbf{2},\mathbf{3}}\mu^{\mathrm{norm}}_u(\mathbf{i})$ with η^*_u, the update rule for summary message $(\eta^1_u,\eta^2_u,\eta^3_u,\eta^*_u)$ in SP is resulted.

■

This theorem suggests that for 3-COL problems, SP *is* PTP. Similar results can be shown for k-SAT problems — instead of showing this result, we will in a later section, show a more general result, namely that weighted SP is weighted PTP for k-SAT problems. It should be convincing then that the general principle of designing SP algorithm for arbitrary CSPs is the recipe specified in the PTP message-update rule.

In the correspondence between SP and PTP for 3-COL problems established in this theorem, it is worth noting that symbol i in the SP messages corresponds to the singleton token **i** that contains the single element i, and symbol $*$ in the SP messages corresponds to the group of all non-singleton tokens. We note that the fact that all non-singleton tokens can be represented by a single symbol $*$ is rather a coincidence, intrinsically related to the structure of 3-COL problems, and should not be understood as a general principle. Specifically, for 3-COL problems, each constraint vertex has degree 2, and as long as a non-singleton token is passed to a constraint vertex, the outgoing token from the constraint vertex will be token **123**. It is precisely due to this fact that all non-singleton tokens can be represented by the same symbol — the joker symbol $*$, as is conventionally termed. This observation then implies that for general CSPs with non-binary alphabet, SP, or equivalently PTP, may be expected to contain more than one "joker" symbol, each corresponding to one or several non-singleton tokens. In other words, this suggests that the notion of "joker" symbol in SP messages is *not* a fundamental one, and that the rather fundamental perspective of SP is the extension of the variable alphabet to its power set with empty set excluded — or equivalently via a one-to-one correspondence, the set of all tokens associated with the variable.

Finally, we remark that there can be a caveat on whether SP and PTP are exactly equivalent, when taking into account the decimation procedure associated with the SP algorithms. Specifically, we note that decimation is performed based on summary messages in SP. For 3-COL problems, each SP summary message contains "biases" on four different symbols, but each PTP summary message contains "biases" on seven different tokens. The natural decimation procedure for PTP is then to fix one "highly biased" variable to one of the seven tokens,

rather than to one of the four symbols. Although it is not clear at this point whether this finer procedure may provide gains in algorithm performance, it nevertheless suggests that PTP is slightly more general than SP. Investigation on possible benefit of this slight generality can be an interesting direction of research.

4.4 Weighted PTP

In the mechanism of passing random tokens that underlies the PTP message-update rule, the outgoing token sent from a variable vertex is a *function* of all incoming tokens from its upstream. A natural angle to generalize the dependency of these outgoing tokens on the incoming tokens is to generalize this *functional dependency* to a *probabilistic dependency*. Specifically, using the "intention-command" analogy, this probabilistic dependency will allow the intention of a variable, conditioned on all incoming commands from the upstream, to take any set of the values — not necessarily the maximal set — that obeys by the commands, and this probabilistic dependency is specified via the probability of each allowed intention. This results in what we call *weighted PTP*.

In weighted PTP, we assume that the token $t_{v\to c}$ passed from variable vertex x_v to constraint vertex Γ_c may be any subset of the intersection of all incoming tokens passed to x_v except that passed from Γ_c, and the probability that token $t_{v\to c}$ equals to each subset is specified via a non-negative function $\omega_v(a|b)$ defined on $(\chi^*)^{\{v\}} \times \Big((\chi^*)^{\{v\}} \cup \{\emptyset_v\}\Big)$ for each $v \in V$. We will restrict $\omega_v(a|b)$ to an *obedience conditional* on $(\chi^*)^{\{v\}}$, the definition of which is given as follows.

Definition 1 (Obedience Conditional) *A non-negative function $h(a|b)$ on $(\chi^*)^{\{v\}}\times\Big((\chi^*)^{\{v\}} \cup \{\emptyset_v\}\Big)$ is said to be an obedience conditional on $(\chi^*)^{\{v\}}$ if $h(a|\emptyset_v) = 0$ for all $a \in (\chi^*)^{\{v\}}$ and $h(a|b) = 0$ for any $a, b \in (\chi^*)^{\{v\}}$ with $a \not\subseteq b$.*

First we note that in the definition, variable a in $h(\cdot)$ is intended to refer to an "intention", variable b is intended to refer to a "command", and function h is evaluated to zero if the command is null or if the intention does not obey the command. This is the reason for which

we name such a function an "obedience" conditional. Second, it is also worth noting that an obedience conditional h as defined above is not a true conditional distribution, since it is not the case that $\sum_a h(a|b) = 1$ for all b. However, it is a minor technicality to modify the definition of h (without impacting the development of any result in this thesis) so that it is indeed a conditional distribution [1]. Thus for the purpose of this research, one may always regard an obedience conditional as a conditional distribution of an intention given a command.

Apparently, function $[a = b]$ is a special case of obedience conditional, characterizing a special *functional* dependency of intention a on command b, namely that the intention set a is exactly the command set b.

We now give the precise message-update rule of weighted PTP where the only difference with PTP is in left message and summary message.

---

[1] Given an obedience conditional h, we may define a conditional distribution $\tilde{h}(a|b)$. Let Z be $\max_{b \in (\chi^*)^{\{v\}}} \sum_{a \in (\chi^*)^{\{v\}}} h(a|b)$. Let non-negative function $\tilde{h}(a|b)$ on $\left((\chi^*)^{\{v\}} \cup \{\emptyset_v\}\right) \times \left((\chi^*)^{\{v\}} \cup \{\emptyset_v\}\right)$ be defined as follows: $\tilde{h}(a|\emptyset_v) := [a = \emptyset_v]$; $\tilde{h}(\emptyset_v|b) := 1 - \sum_{a \in (\chi^*)^{\{v\}}} h(a|b)/Z$ for all $b \neq \emptyset_v$; and for all other (a, b), $\tilde{h}(a|b) := h(a|b)/Z$. It is easy to see that $\tilde{h}(a|b)$ is a conditional distribution. Since eventually we will condition on that $a \neq \emptyset$, it is straight-forward to verify that the role of h is equivalent to $\tilde{h}$.

Weighted PTP Message-Update Rule

$$\lambda_{v\to c}(t_{v\to c}) := \sum_{\langle t_{b\to v}\rangle_{b\in C(v)\setminus\{c\}}} \omega_v\left(t_{v\to c}\middle|\bigcap_{b\in C(v)\setminus\{c\}} t_{b\to v}\right)\prod_{b\in C(v)\setminus\{c\}}\rho^{\text{norm}}_{b\to v}(t_{b\to v}) \quad (4.33)$$

$$\rho_{c\to v}(t_{c\to v}) := \sum_{\langle t_{u\to c}\rangle_{u\in V(c)\setminus\{v\}}}\left[t_{c\to v}=\mathtt{F}_c\left(\prod_{u\in V(c)\setminus\{v\}} t_{u\to c}\right)\right]\prod_{u\in V(c)\setminus\{v\}}\lambda^{\text{norm}}_{u\to c}(t_{u\to c}) \quad (4.34)$$

$$\mu_v(t_v) := \sum_{\langle t_{c\to v}\rangle_{c\in C(v)}} \omega_v\left(t_v\middle|\bigcap_{c\in C(v)} t_{c\to v}\right)\prod_{c\in C(v)}\rho^{\text{norm}}_{c\to v}(t_{c\to v}), \quad (4.35)$$

and the normalized messages are defined as

$$\lambda^{\text{norm}}_{v\to c}(t_{v\to c}) := \lambda_{v\to c}(t_{v\to c})/\sum_{t\in(\chi^*)^{\{v\}}}\lambda_{v\to c}(t) \quad (4.36)$$

$$\rho^{\text{norm}}_{c\to v}(t_{c\to v}) := \rho_{c\to v}(t_{c\to v})/\sum_{t\in(\chi^*)^{\{v\}}}\rho_{c\to v}(t) \quad (4.37)$$

$$\mu^{\text{norm}}_{v}(t_v) := \mu_v(t_v)/\sum_{t\in(\chi^*)^{\{v\}}}\mu_v(t). \quad (4.38)$$

It is easily seen that weighted PTP is a family of algorithms, parametrized by a collection of obedience conditionals, $\{\omega_v : v \in V\}$, each for a coordinate. The fact that conditional distribution $\omega_v(a|b)$ generalizes indicator function $[a = b]$ immediately implies that weighted PTP generalizes PTP, as stated in the following lemma.

Lemma 7 *If $\omega_v(a|b) := [a = b]$ for all $v \in V$, then weighted PTP is PTP.*

4.4.1 Weighted PTP Generalizes Weighted SP

Now we will show that the weighted SP developed for k-SAT problems [23] is a special case of weighted PTP. That is, for k-SAT problems, when setting functions $\{\omega_v : v \in V\}$ in weighted PTP to a particular form, weighted SP, or SP$^*(\epsilon)$ is resulted.

For a k-SAT problem, let function $\omega_v(a|b)$ for every $v \in V$ in weighted PTP be defined via a single real number $\epsilon \in [0,1]$ as follows.

$$\omega_v(a|b) := \begin{cases} \epsilon, & \text{if } a = b = \mathbf{01} \\ 1-\epsilon, & \text{if } a \subset b = \mathbf{01} \\ 1, & \text{if } a = b \neq \mathbf{01} \\ 0, & \text{otherwise} \end{cases} \tag{4.39}$$

Lemma 8 *Let $\{\omega_v : v \in V\}$ in k-SAT be defined as in (4.39). The message-update rule of weighted PTP is then:*

$$\lambda_{v\to c}(\mathbf{0}) := \prod_{b\in C(v)\setminus\{c\}} (\rho^{\text{norm}}_{b\to v}(\mathbf{0}) + \rho^{\text{norm}}_{b\to v}(\mathbf{01})) - \epsilon \prod_{b\in C(v)\setminus\{c\}} \rho^{\text{norm}}_{b\to v}(\mathbf{01}) \tag{4.40}$$

$$\lambda_{v\to c}(\mathbf{1}) := \prod_{b\in C(v)\setminus\{c\}} (\rho^{\text{norm}}_{b\to v}(\mathbf{1}) + \rho^{\text{norm}}_{b\to v}(\mathbf{01})) - \epsilon \prod_{b\in C(v)\setminus\{c\}} \rho^{\text{norm}}_{b\to v}(\mathbf{01}) \tag{4.41}$$

$$\lambda_{v\to c}(\mathbf{01}) := \epsilon \prod_{b\in C(v)\setminus\{c\}} \rho^{\text{norm}}_{b\to v}(\mathbf{01}) \tag{4.42}$$

$$\rho_{c\to v}(\mathbf{0}) := [L_{v,c} = 0] \cdot \prod_{u\in V(c)\setminus\{v\}: L_{u,c}=1} \lambda^{\text{norm}}_{u\to c}(\mathbf{0}) \cdot \prod_{u\in V(c)\setminus\{v\}: L_{u,c}=0} \lambda^{\text{norm}}_{u\to c}(\mathbf{1}) \tag{4.43}$$

$$\rho_{c\to v}(\mathbf{1}) := [L_{v,c} = 1] \cdot \prod_{u\in V(c)\setminus\{v\}: L_{u,c}=1} \lambda^{\text{norm}}_{u\to c}(\mathbf{0}) \cdot \prod_{u\in V(c)\setminus\{v\}: L_{u,c}=0} \lambda^{\text{norm}}_{u\to c}(\mathbf{1}) \tag{4.44}$$

$$\rho_{c\to v}(\mathbf{01}) := 1 - \prod_{\substack{u\in V(c)\setminus\{v\}: \\ L_{u,c}=1}} \lambda^{\text{norm}}_{u\to c}(\mathbf{0}) \cdot \prod_{\substack{u\in V(c)\setminus\{v\}: \\ L_{u,c}=0}} \lambda^{\text{norm}}_{u\to c}(\mathbf{1}) \tag{4.45}$$

$$\mu_v(\mathbf{0}) := \prod_{c\in C(v)} (\rho^{\text{norm}}_{c\to v}(\mathbf{0}) + \rho^{\text{norm}}_{c\to v}(\mathbf{01})) - \epsilon \prod_{c\in C(v)} \rho^{\text{norm}}_{c\to v}(\mathbf{01}) \tag{4.46}$$

$$\mu_v(\mathbf{1}) := \prod_{c\in C(v)} (\rho^{\text{norm}}_{c\to v}(\mathbf{1}) + \rho^{\text{norm}}_{c\to v}(\mathbf{01})) - \epsilon \prod_{c\in C(v)} \rho^{\text{norm}}_{c\to v}(\mathbf{01}) \tag{4.47}$$

$$\mu_v(\mathbf{01}) := \epsilon \prod_{c\in C(v)} \rho^{\text{norm}}_{c\to v}(\mathbf{01}). \tag{4.48}$$

Proof: These update equations can be immediately obtained from weighted PTP message

update equations (4.33) to (4.35), where (4.45) follows from

$$\begin{aligned}\rho_{c\to v}(\mathbf{01}) &= \prod_{u\in V(c)\setminus\{v\}} (\lambda^{\text{norm}}_{u\to c}(\mathbf{0}) + \lambda^{\text{norm}}_{u\to c}(\mathbf{1}) + \lambda^{\text{norm}}_{u\to c}(\mathbf{01})) - \prod_{\substack{u\in V(c)\setminus\{v\}:\\ L_{u,c}=1}} \lambda^{\text{norm}}_{u\to c}(\mathbf{0}) \cdot \prod_{\substack{u\in V(c)\setminus\{v\}:\\ L_{u,c}=0}} \lambda^{\text{norm}}_{u\to c}(\mathbf{1}) \\ &= 1 - \prod_{\substack{u\in V(c)\setminus\{v\}:\\ L_{u,c}=1}} \lambda^{\text{norm}}_{u\to c}(\mathbf{0}) \cdot \prod_{\substack{u\in V(c)\setminus\{v\}:\\ L_{u,c}=0}} \lambda^{\text{norm}}_{u\to c}(\mathbf{1}).\end{aligned}$$

■

Theorem 2 *Let $\{\omega_v : v \in V\}$ in a k-SAT problem be defined as in (4.39). Denote by $(\Pi^{\mathtt{s}\ \text{norm}}_{v\to c}, \Pi^{\mathtt{u}\ \text{norm}}_{v\to c}, \Pi^{*\ \text{norm}}_{v\to c})$ the normalized version of SP message $(\Pi^{\mathtt{s}}_{v\to c}, \Pi^{\mathtt{u}}_{v\to c}, \Pi^{*}_{v\to c})$, namely that $\Pi^{\mathtt{s}\ \text{norm}}_{v\to c} = \Pi^{\mathtt{s}}_{v\to c}/\left(\Pi^{\mathtt{s}}_{v\to c} + \Pi^{\mathtt{u}}_{v\to c} + \Pi^{*}_{v\to c}\right)$, $\Pi^{\mathtt{u}\ \text{norm}}_{v\to c} = \Pi^{\mathtt{u}}_{v\to c}/\left(\Pi^{\mathtt{s}}_{v\to c} + \Pi^{\mathtt{u}}_{v\to c} + \Pi^{*}_{v\to c}\right)$, and $\Pi^{*\ \text{norm}}_{v\to c} = \Pi^{\mathtt{s}}_{v\to c}/\left(\Pi^{\mathtt{s}}_{v\to c} + \Pi^{\mathtt{u}}_{v\to c} + \Pi^{*}_{v\to c}\right)$. Then the correspondence between SP*(ϵ) message-update rule and weighted PTP message-update rule is*

$$\Pi^{\mathtt{s}\ \text{norm}}_{v\to c} \leftrightarrow [L_{v,c}=0]\cdot\lambda^{\text{norm}}_{v\to c}(\mathbf{0}) + [L_{v,c}=1]\cdot\lambda^{\text{norm}}_{v\to c}(\mathbf{1}) \tag{4.49}$$

$$\Pi^{\mathtt{u}\ \text{norm}}_{v\to c} \leftrightarrow [L_{v,c}=0]\cdot\lambda^{\text{norm}}_{v\to c}(\mathbf{1}) + [L_{v,c}=1]\cdot\lambda^{\text{norm}}_{v\to c}(\mathbf{0}) \tag{4.50}$$

$$\Pi^{*\ \text{norm}}_{v\to c} \leftrightarrow \lambda^{\text{norm}}_{v\to c}(\mathbf{01}) \tag{4.51}$$

$$\eta_{c\to v} \leftrightarrow \rho^{\text{norm}}_{c\to v}(\mathbf{0}) + \rho^{\text{norm}}_{c\to v}(\mathbf{1}) \tag{4.52}$$

$$\zeta^0_v \leftrightarrow \mu_v(\mathbf{0}) \tag{4.53}$$

$$\zeta^1_v \leftrightarrow \mu_v(\mathbf{1}) \tag{4.54}$$

$$\zeta^*_v \leftrightarrow \mu_v(\mathbf{01}). \tag{4.55}$$

Prior to proving the theorem, we will introduce some notations and a simple lemma which will be useful in the proof. For any neighboring variable vertex x_v and constraint vertex Γ_c, we will denote by $\mathbf{L_{v,c}}$ the singleton token containing the single elementary assignment that assigns coordinate v the edge label $L_{v,c}$. Similarly, we will denote by $\bar{\mathbf{L}}_{\mathbf{v,c}}$ the singleton token containing the single elementary assignment that assigns coordinate v the negated edge label $\bar{L}_{v,c}$. With these notations, the following lemma immediately follows from Lemma 8.

Lemma 9 *For any $(v-c)$ pair in a k-SAT problem, the right message $\rho^{\text{norm}}_{c\to v}$ satisfies:*

$$\rho^{\text{norm}}_{c\to v}(\mathbf{L_{v,c}}) + \rho^{\text{norm}}_{c\to v}(\mathbf{01}) = 1 \tag{4.56}$$

$$\rho^{\text{norm}}_{c\to v}(\bar{\mathbf{L}}_{\mathbf{v,c}}) + \rho^{\text{norm}}_{c\to v}(\mathbf{01}) = \rho^{\text{norm}}_{c\to v}(\mathbf{01}). \tag{4.57}$$

Now we are ready to prove Theorem 2.

Proof: We will refer to the message correspondence in Equations (4.49) to (4.51) as the "left correspondence", the correspondence in (4.52) as the "right correspondence", and the correspondence in Equations (4.53) to (4.55) as the "summary correspondence".

We will prove the theorem by first showing that if the left correspondence holds, then the right correspondence holds, and conversely that if the right correspondence holds, then the left correspondence holds. This should prove that correspondence between $\text{SP}^*(\epsilon)$ and weighted PTP in their passed messages. We will then complete the proof by showing the summary correspondence.

First suppose that the left correspondence holds, namely that $\Pi^{\mathbf{s}\ \text{norm}}_{v\to c} = [L_{v,c}=0]\cdot\lambda^{\text{norm}}_{v\to c}(\mathbf{0}) + [L_{v,c}=1]\cdot\lambda^{\text{norm}}_{v\to c}(\mathbf{1})$, $\Pi^{\mathbf{u}\ \text{norm}}_{v\to c} = [L_{v,c}=0]\cdot\lambda^{\text{norm}}_{v\to c}(\mathbf{1}) + [L_{v,c}=1]\cdot\lambda^{\text{norm}}_{v\to c}(\mathbf{0})$, and $\Pi^{*\ \text{norm}}_{v\to c} = \lambda^{\text{norm}}_{v\to c}(\mathbf{01})$.

In each iteration, by Lemma 8 and the fact $[L_{v,c}=1]+[L_{v,c}=0]=1$ for every $(v-c)$ pair, the right messages satisfy

$$\begin{aligned}
\rho_{c\to v}(\mathbf{0}) + \rho_{c\to v}(\mathbf{1}) + \rho_{c\to v}(\mathbf{01}) \quad &= [L_{v,c}=0]\cdot \prod_{u\in V(c)\setminus\{v\}:L_{u,c}=1}\lambda^{\text{norm}}_{u\to c}(\mathbf{0})\cdot \prod_{u\in V(c)\setminus\{v\}:L_{u,c}=0}\lambda^{\text{norm}}_{u\to c}(\mathbf{1}) \\
&\quad +[L_{v,c}=1]\cdot \prod_{u\in V(c)\setminus\{v\}:L_{u,c}=1}\lambda^{\text{norm}}_{u\to c}(\mathbf{0})\cdot \prod_{u\in V(c)\setminus\{v\}:L_{u,c}=0}\lambda^{\text{norm}}_{u\to c}(\mathbf{1}) \\
&\quad +1 - \prod_{u\in V(c)\setminus\{v\}:L_{u,c}=1}\lambda^{\text{norm}}_{u\to c}(\mathbf{0})\cdot \prod_{u\in V(c)\setminus\{v\}:L_{u,c}=0}\lambda^{\text{norm}}_{u\to c}(\mathbf{1}) \\
&= 1.
\end{aligned}$$

That is, each right message $\rho_{c\to v}$ is already normalized, or $\rho_{c\to v} = \rho^{\text{norm}}_{c\to v}$. Then

$$\begin{aligned}
\rho^{\text{norm}}_{c\to v}(\mathbf{0}) + \rho^{\text{norm}}_{c\to v}(\mathbf{1}) &= \rho_{c\to v}(\mathbf{0}) + \rho_{c\to v}(\mathbf{1}) \\
&= [L_{v,c}=0] \cdot \prod_{u\in V(c)\setminus\{v\}:L_{u,c}=1} \lambda^{\text{norm}}_{u\to c}(\mathbf{0}) \cdot \prod_{u\in V(c)\setminus\{v\}:L_{u,c}=0} \lambda^{\text{norm}}_{u\to c}(\mathbf{1}) \\
&\quad + [L_{v,c}=1] \cdot \prod_{u\in V(c)\setminus\{v\}:L_{u,c}=1} \lambda^{\text{norm}}_{u\to c}(\mathbf{0}) \cdot \prod_{u\in V(c)\setminus\{v\}:L_{u,c}=0} \lambda^{\text{norm}}_{u\to c}(\mathbf{1}) \\
&= \prod_{u\in V(c)\setminus\{v\}:L_{u,c}=1} \lambda^{\text{norm}}_{u\to c}(\mathbf{0}) \cdot \prod_{u\in V(c)\setminus\{v\}:L_{u,c}=0} \lambda^{\text{norm}}_{u\to c}(\mathbf{1}) \\
&= \prod_{u\in V(c)\setminus\{v\}:L_{u,c}=1} ([L_{u,c}=1]\cdot\lambda^{\text{norm}}_{u\to c}(\mathbf{0}) + [L_{u,c}=0]\cdot\lambda^{\text{norm}}_{u\to c}(\mathbf{1})) \\
&\quad \cdot \prod_{u\in V(c)\setminus\{v\}:L_{u,c}=0} ([L_{u,c}=1]\cdot\lambda^{\text{norm}}_{u\to c}(\mathbf{0}) + [L_{u,c}=0]\cdot\lambda^{\text{norm}}_{u\to c}(\mathbf{1})) \\
&= \prod_{u\in V(c)\setminus\{v\}} ([L_{u,c}=1]\cdot\lambda^{\text{norm}}_{u\to c}(\mathbf{0}) + [L_{u,c}=0]\cdot\lambda^{\text{norm}}_{u\to c}(\mathbf{1})) \\
&\overset{(a)}{=} \prod_{u\in V(c)\setminus\{v\}} \Pi^{\mathtt{u}\ \text{norm}}_{u\to c} \\
&\overset{(b)}{=} \prod_{u\in V(c)\setminus\{v\}} \frac{\Pi^{\mathtt{u}}_{u\to c}}{\Pi^{\mathtt{u}}_{u\to c} + \Pi^{\mathtt{s}}_{u\to c} + \Pi^{*}_{u\to c}} \\
&= \eta_{c\to v},
\end{aligned}$$

where equality (a) is due to the assumed left correspondence, and equality (b) follows from the definition of $\Pi^{\mathtt{u}\ \text{norm}}_{v\to c}$. Thus we have shown that if the left correspondence holds, then the right correspondence holds.

Now suppose that the right correspondence holds, namely that $\eta_{c\to v} = \rho^{\text{norm}}_{c\to v}(\mathbf{0}) + \rho^{\text{norm}}_{c\to v}(\mathbf{1})$

for every $(v-c)$ pair. Following the PTP message-update equations (4.40) to (4.42), we have

$$
\begin{aligned}
& [L_{v,c}=0]\cdot\lambda_{v\to c}(\mathbf{0})+[L_{v,c}=1]\cdot\lambda_{v\to c}(\mathbf{1}) \\
= \quad & [L_{v,c}=0]\cdot\left(\prod_{b\in C(v)\backslash\{c\}}(\rho^{\text{norm}}_{b\to v}(\mathbf{0})+\rho^{\text{norm}}_{b\to v}(\mathbf{01}))-\epsilon\prod_{b\in C(v)\backslash\{c\}}\rho^{\text{norm}}_{b\to v}(\mathbf{01})\right) \\
& +[L_{v,c}=1]\cdot\left(\prod_{b\in C(v)\backslash\{c\}}(\rho^{\text{norm}}_{b\to v}(\mathbf{1})+\rho^{\text{norm}}_{b\to v}(\mathbf{01}))-\epsilon\prod_{b\in C(v)\backslash\{c\}}\rho^{\text{norm}}_{b\to v}(\mathbf{01})\right) \\
= \quad & [L_{v,c}=0]\cdot\prod_{b\in C(v)\backslash\{c\}}(\rho^{\text{norm}}_{b\to v}(\mathbf{0})+\rho^{\text{norm}}_{b\to v}(\mathbf{01}))+[L_{v,c}=1]\cdot\prod_{b\in C(v)\backslash\{c\}}(\rho^{\text{norm}}_{b\to v}(\mathbf{1})+\rho^{\text{norm}}_{b\to v}(\mathbf{01})) \\
& -\epsilon\prod_{b\in C(v)\backslash\{c\}}\rho^{\text{norm}}_{b\to v}(\mathbf{01}) \\
\overset{(4.57)}{=} \quad & [L_{v,c}=0]\cdot\prod_{b\in C^{\text{s}}_c(v)}(\rho^{\text{norm}}_{b\to v}(\mathbf{0})+\rho^{\text{norm}}_{b\to v}(\mathbf{01}))\cdot\prod_{b\in C^{\text{u}}_c(v)}\rho^{\text{norm}}_{b\to v}(\mathbf{01}) \\
& +[L_{v,c}=1]\cdot\prod_{b\in C^{\text{s}}_c(v)}(\rho^{\text{norm}}_{b\to v}(\mathbf{1})+\rho^{\text{norm}}_{b\to v}(\mathbf{01}))\cdot\prod_{b\in C^{\text{u}}_c(v)}\rho^{\text{norm}}_{b\to v}(\mathbf{01})-\epsilon\prod_{b\in C(v)\backslash\{c\}}\rho^{\text{norm}}_{b\to v}(\mathbf{01}) \\
\overset{(4.56)}{=} \quad & [L_{v,c}=0]\cdot\prod_{b\in C^{\text{u}}_c(v)}\rho^{\text{norm}}_{b\to v}(\mathbf{01})+[L_{v,c}=1]\cdot\prod_{b\in C^{\text{u}}_c(v)}\rho^{\text{norm}}_{b\to v}(\mathbf{01})-\epsilon\prod_{b\in C(v)\backslash\{c\}}\rho^{\text{norm}}_{b\to v}(\mathbf{01}) \\
= \quad & \prod_{b\in C^{\text{u}}_c(v)}\rho^{\text{norm}}_{b\to v}(\mathbf{01})-\epsilon\prod_{b\in C(v)\backslash\{c\}}\rho^{\text{norm}}_{b\to v}(\mathbf{01}) \\
= \quad & \prod_{b\in C^{\text{u}}_c(v)}\rho^{\text{norm}}_{b\to v}(\mathbf{01})\cdot\left(1-\epsilon\prod_{b\in C^{\text{s}}_c(v)}\rho^{\text{norm}}_{b\to v}(\mathbf{01})\right) \\
= \quad & \prod_{b\in C^{\text{u}}_c(v)}(1-\rho^{\text{norm}}_{b\to v}(\mathbf{0})-\rho^{\text{norm}}_{b\to v}(\mathbf{1}))\cdot\left(1-\epsilon\prod_{b\in C^{\text{s}}_c(v)}(1-\rho^{\text{norm}}_{b\to v}(\mathbf{0})-\rho^{\text{norm}}_{b\to v}(\mathbf{1}))\right) \\
\overset{(c)}{=} \quad & \prod_{b\in C^{\text{u}}_c(v)}(1-\eta_{b\to v})\cdot\left(1-\epsilon\prod_{b\in C^{\text{s}}_c(v)}(1-\eta_{b\to v})\right)=\Pi^{\text{s}}_{v\to c},
\end{aligned}
$$

where equality (c) above is due to the assumed right correspondence. We will denote this result by (A).

Following very similar procedures, it can be shown that

$$\begin{aligned} & [L_{v,c}=0]\cdot\lambda_{v\to c}(\mathbf{1})+[L_{v,c}=1]\cdot\lambda_{v\to c}(\mathbf{0}) \\ = & \prod_{b\in C_c^{\mathtt{s}}(v)}(1-\rho_{b\to v}^{\mathrm{norm}}(\mathbf{0})-\rho_{b\to v}^{\mathrm{norm}}(\mathbf{1}))\cdot\left(1-\epsilon\prod_{b\in C_c^{\mathtt{u}}(v)}(1-\rho_{b\to v}^{\mathrm{norm}}(\mathbf{0})-\rho_{b\to v}^{\mathrm{norm}}(\mathbf{1}))\right) \\ = & \Pi_{v\to c}^{\mathtt{u}} \end{aligned}$$

We will denote this result by (B).

Similarly,

$$\begin{aligned} \lambda_{v\to c}(\mathbf{01}) &= \epsilon\prod_{b\in C_c^{\mathtt{s}}(v)}(1-\rho_{b\to v}^{\mathrm{norm}}(\mathbf{0})-\rho_{b\to v}^{\mathrm{norm}}(\mathbf{1}))\cdot\prod_{b\in C_c^{\mathtt{u}}(v)}(1-\rho_{b\to v}^{\mathrm{norm}}(\mathbf{0})-\rho_{b\to v}^{\mathrm{norm}}(\mathbf{1})) \\ &= \Pi_{v\to c}^{*}. \end{aligned}$$

We will denote this result by (C).

Combining results (A), (B) and (C), we have

$$\lambda_{v\to c}(\mathbf{0})+\lambda_{v\to c}(\mathbf{1})+\lambda_{v\to c}(\mathbf{01})=\Pi_{v\to c}^{\mathtt{u}}+\Pi_{v\to c}^{\mathtt{s}}+\Pi_{v\to c}^{*}.$$

That is, the scaling constant for normalizing $(\lambda_{v\to c}(\mathbf{0}),\lambda_{v\to c}(\mathbf{1}),\lambda_{v\to c}(\mathbf{01}))$ and that for normalizing $(\Pi_{v\to c}^{\mathtt{u}},\Pi_{v\to c}^{\mathtt{s}},\Pi_{v\to c}^{*})$ are identical. Then results (A), (B) and (C) respectively translate to

$$\begin{aligned} [L_{v,c}=1]\cdot\lambda_{v\to c}^{\mathrm{norm}}(\mathbf{1})+[L_{v,c}=0]\cdot\lambda_{v\to c}^{\mathrm{norm}}(\mathbf{0}) &= \Pi_{v\to c}^{\mathtt{s}\ \mathrm{norm}} \\ [L_{v,c}=0]\cdot\lambda_{v\to c}^{\mathrm{norm}}(\mathbf{1})+[L_{v,c}=1]\cdot\lambda_{v\to c}^{\mathrm{norm}}(\mathbf{0}) &= \Pi_{v\to c}^{\mathtt{u}\ \mathrm{norm}} \\ \lambda_{v\to c}^{\mathrm{norm}}(\mathbf{01}) &= \Pi_{v\to c}^{*\ \mathrm{norm}}. \end{aligned}$$

At this point we have established the correspondence between the passed messages in

weighted PTP and those in weighted SP. We now prove the summary correspondence.

Starting from Lemma 8, we have

$$\begin{aligned}
\mu_v(\mathbf{0}) &= \prod_{c\in C(v)} (\rho^{\text{norm}}_{c\to v}(\mathbf{0}) + \rho^{\text{norm}}_{c\to v}(\mathbf{01})) - \epsilon \prod_{c\in C(v)} \rho^{\text{norm}}_{c\to v}(\mathbf{01}) \\
&= \prod_{c\in C^1(v)} (\rho^{\text{norm}}_{c\to v}(\mathbf{0}) + \rho^{\text{norm}}_{c\to v}(\mathbf{01})) \prod_{c\in C^0(v)} (\rho^{\text{norm}}_{c\to v}(\mathbf{0}) + \rho^{\text{norm}}_{c\to v}(\mathbf{01})) - \epsilon \prod_{c\in C(v)} \rho^{\text{norm}}_{c\to v}(\mathbf{01}) \\
&\overset{(4.56),(4.57)}{=} \prod_{c\in C^1(v)} \rho^{\text{norm}}_{c\to v}(\mathbf{01}) - \epsilon \prod_{c\in C(v)} \rho^{\text{norm}}_{c\to v}(\mathbf{01}) \\
&= \left(1 - \epsilon \prod_{c\in C^0(v)} \rho^{\text{norm}}_{c\to v}(\mathbf{01})\right) \prod_{c\in C^1(v)} \rho^{\text{norm}}_{c\to v}(\mathbf{01}) \\
&= \left(1 - \epsilon \prod_{c\in C^0(v)} (1 - \rho^{\text{norm}}_{c\to v}(\mathbf{0}) - \rho^{\text{norm}}_{c\to v}(\mathbf{1}))\right) \prod_{c\in C^1(v)} (1 - \rho^{\text{norm}}_{c\to v}(\mathbf{0}) - \rho^{\text{norm}}_{c\to v}(\mathbf{1})) \\
&\overset{(d)}{=} \left(1 - \epsilon \prod_{c\in C^0(v)} (1 - \eta_{c\to v})\right) \prod_{c\in C^1(v)} (1 - \eta_{c\to v})) \\
&= \zeta^0_v
\end{aligned}$$

where (d) above is due to the right correspondence that we just proved.

Symmetrically, it can be shown that

$$\begin{aligned}
\mu_v(\mathbf{1}) &= \left(1 - \epsilon \prod_{c\in C^1(v)} (1 - \rho^{\text{norm}}_{c\to v}(\mathbf{0}) - \rho^{\text{norm}}_{c\to v}(\mathbf{1}))\right) \prod_{c\in C^0(v)} (1 - \rho^{\text{norm}}_{c\to v}(\mathbf{0}) - \rho^{\text{norm}}_{c\to v}(\mathbf{1})) \\
&= \left(1 - \epsilon \prod_{c\in C^1(v)} (1 - \eta_{c\to v})\right) \prod_{c\in C^0(v)} (1 - \eta_{c\to v}) \\
&= \zeta^1_v.
\end{aligned}$$

Finally, it is straight-forward to see

$$\begin{aligned}
\mu_v(\mathbf{01}) &= \epsilon \prod_{c\in C^0(v)} (1-\rho^{\text{norm}}_{c\to v}(\mathbf{0})-\rho^{\text{norm}}_{c\to v}(\mathbf{1})) \prod_{c\in C^1(v)} (1-\rho^{\text{norm}}_{c\to v}(\mathbf{0})-\rho^{\text{norm}}_{c\to v}(\mathbf{1})) \\
&= \epsilon \prod_{c\in C^0(v)} (1-\eta_{c\to v}) \prod_{c\in C^1(v)} (1-\eta_{c\to v}) \\
&= \zeta^*_v.
\end{aligned}$$

This proves the summary correspondence and completes the proof. ■

This theorem asserts that weighted SP developed for k-SAT problems is an instance of weighted PTP that we propose in this proposal, or alternatively phrased, weighted PTP generalizes weighted SP from the context of k-SAT problems to arbitrary CSPs with arbitrary variable alphabets. When specifying parameter ϵ to be 1, this result immediately implies that non-weighted SP is non-weighted PTP for k-SAT problems.

Additionally, we note that in the correspondence between the summary messages of weighted PTP and weighted SP in the above theorem, it is clear that symbols $0, 1$, and $*$ in weighted SP (or SP) corresponds to tokens (sets) $\mathbf{0}$, $\mathbf{1}$ and $\mathbf{01}$ respectively. In addition, if we use notation $\mathbf{L_{v,c}}$, we may re-write the correspondence between the left messages of weighted SP and those of weighted PTP in the above theorem as

$$\begin{aligned}
\Pi^{\mathtt{s}}_{v\to c} &\leftrightarrow \lambda_{v\to c}(\mathbf{L_{v,c}}) \\
\Pi^{\mathtt{u}}_{v\to c} &\leftrightarrow \lambda_{v\to c}(\bar{\mathbf{L}}_{\mathbf{v,c}}) \\
\Pi^{*}_{v\to c} &\leftrightarrow \lambda_{v\to c}(\mathbf{01})
\end{aligned}$$

That is, symbols "$\mathtt{s}$" and "$\mathtt{u}$" in SP respectively correspond to singleton set $\mathbf{L_{v,c}}$ and $\bar{\mathbf{L}}_{\mathbf{v,c}}$. These observations suggest that, although blurred by the addition of single symbol $*$ to the variable alphabet, the true alphabet used as the support of SP messages is the set of all tokens associated with the variable, or equivalently, the power set of the original alphabet with the

empty set removed.

Finally, we would like to note that the generalization of PTP to weighted PTP (or from SP to weighted SP) entails only a slight increase of computation complexity. More specifically, the computation of messages in weighted PTP requires computing probability mass functions of larger support. This increase of complexity can be loosely upper-bounded by the constant scaling factor $2^{|\mathcal{X}|}$. Such a linear increase of complexity for relatively small alphabets is typically negligible, as the main complexity concerns are with respect to the problem size, namely the number of variables involved.

4.5 On the Dynamics of SP

We now present some results concerning the dynamics of SP, based on the formulation of PTP and weighted PTP. These results, although rather elementary, should help provide intuitions regarding what PTP is doing in solving a CSP. We will start with the deterministic precursor of PTP, DTP.

4.5.1 On the Dynamics of DTP

We will refer to a subgraph H of factor graph G as a *factor-subgraph* of G if for every constraint vertex Γ_c in H, all neighboring variable vertices of Γ_c in G are also in H. It is apparent that factor-subgraph H is a factor graph representing a CSP involving precisely a subset of the constraints in G. We will denote by $C[H]$ the index set of all constraint vertices in H, by $V[H]$ the index set of all variable vertices in H, and by Γ_H the set of all assignments on $V[H]$ that satisfy every constraint Γ_c, $c \in C[H]$.

If factor-subgraph H is a tree, it is also referred to as a *factor tree* of G. For any factor tree T of G, we will denote by $L[T]$ the index set of all leaf vertices of T. Since we have assumed that factor graph G contains no degree-1 constraint vertices, it is necessary that the leaf vertices of any factor tree T of G are all variable vertices, i.e., that $L[T]$ contains no index

of any constraint vertex.

Suppose that T is a factor tree of factor graph G, $U \subset V[T]$, and $v \in V[T] \setminus U$. For any rectangle t_U on U, define

$$\mathtt{F}_T^{U \to v}(t_U) := \left(\left(t_U \times \chi^{V[T] \setminus U} \right) \cap \Gamma_T \right)_{:\{v\}}.$$

It is easy to see that function $\mathtt{F}_T^{U \to v}(\cdot)$ reduces to $\mathtt{F}_c^v(\cdot)$ introduced earlier, when T contains a single factor and U is $V(c) \setminus \{v\}$.

Given a factor tree T of G and two vertices in T indexed by a and b respectively, we will introduce another notation of message index, $a \xrightarrow{T} b$, which indexes the message sent by the vertex with index a along its only edge that is on the path (in T) leading to the vertex with index b. For example, suppose that in factor tree T, constraint vertex Γ_c has a neighbor of x_u and is on the path from x_u to x_v in T, then message index $u \xrightarrow{T} v$ is equivalent to $u \to c$, and $t_{u \xrightarrow{T} v}$ is equivalent to $t_{u \to c}$.

A factor tree T of G will be referred to as a (v, l)-tree of G if the variable vertex x_v is in T, every leaf vertex in T is distance $2l$ from vertex x_v, and all vertices in G that have distance to x_v no larger than $2l$ are contained in T. It is clear that given G, $v \in V$ and a positive integer l, if a (v, l)-tree of G exists, it is unique. We therefore denote it by T_v^l.

Given T_v^l of factor graph G, factor tree T_{v-c}^l of G is the subgraph of T_v^l induced by vertex x_v and all vertices of T_v^l whose paths to x_v (in T_v^l) traverse through vertex Γ_c. On the other hand, factor tree $T_{v \neq c}^l$ is the subgraph of T_v^l induced by vertex x_v and all vertices of T_v^l whose paths to x_v (in T_v^l) *do not* traverse through vertex Γ_c.

In what follows, we will use superscript (l) on a message to refer to the message in the l^{th} iteration.

Proposition 1 *Suppose that $l \geq 1$ and that factor tree T_v^l of factor graph G exists. Then in*

iteration l of DTP,

$$t^{(l)}_{c\to v} = \mathtt{F}^{L[T^l_{v-c}]\to v}_{T^l_{v-c}} \left(\prod_{u\in L[T^l_{v-c}]} t^{(1)}_{u \overset{T^l_{v-c}}{\longrightarrow} v} \right).$$

Proof: We will prove this result by induction on l.

For the base case, we have

$$\begin{aligned} t^{(1)}_{c\to v} &= \mathtt{F}^v_c \left(\prod_{u\in V(c)\setminus\{v\}} t^{(1)}_{u\to c} \right) \\ &= \mathtt{F}^{L[T^1_{v-c}]\to v}_{T^1_{v-c}} \left(\prod_{u\in L[T^1_{v-c}]} t^{(1)}_{u \overset{T^1_{v-c}}{\longrightarrow} v} \right). \end{aligned}$$

As the inductive hypothesis, suppose that the result of this proposition holds for a given iteration number $l \geq 1$. This implies specifically that for every $u \in V(c) \setminus \{v\}$ and every $b \in C(u) \setminus \{c\}$,

$$t^{(l)}_{b\to u} = \mathtt{F}^{L[T^l_{u-b}]\to u}_{T^l_{u-b}} \left(\prod_{w\in L[T^l_{u-b}]} t^{(1)}_{w \overset{T^l_{u-b}}{\longrightarrow} u} \right).$$

Then

$$\begin{aligned} t^{(l+1)}_{u\to c} &= \bigcap_{b\in C(u)\setminus\{c\}} t^{(l)}_{b\to u} \\ &= \bigcap_{b\in C(u)\setminus\{c\}} \mathtt{F}^{L[T^l_{u-b}]\to u}_{T^l_{u-b}} \left(\prod_{w\in L[T^l_{u-b}]} t^{(1)}_{w \overset{T^l_{u-b}}{\longrightarrow} u} \right) \\ &= \mathtt{F}^{L[T^l_{u\neq c}]\to u}_{T^l_{u\neq c}} \left(\prod_{w\in L[T^l_{u\neq c}]} t^{(1)}_{w \overset{T^l_{u\neq c}}{\longrightarrow} u} \right). \end{aligned}$$

Finally,

$$\begin{aligned} t_{c\to v}^{(l+1)} &= \mathtt{F}_c^v \left(\prod_{u\in V(c)\setminus\{v\}} t_{u\to c}^{(l+1)} \right) \\ &= \mathtt{F}_c^v \left(\prod_{u\in V(c)\setminus\{v\}} \mathtt{F}_{T_{u\neq c}^l}^{L[T_{u\neq c}^l]\to u} \left(\prod_{w\in L[T_{u\neq c}^l]} t^{(1)}_{T_{u\neq c}^l \; w\xrightarrow{} u} \right) \right) \\ &= \mathtt{F}_{T_{v-c}^{l+1}}^{L[T_{v-c}^{l+1}]\to v} \left(\prod_{w\in L[T_{v-c}^{l+1}]} t^{(1)}_{T_{v-c}^{l+1} \; w\xrightarrow{} v} \right). \end{aligned}$$

This completes the proof. ■

Translating this results to summary tokens, the following result can be obtained immediately.

Corollary 1 *Suppose that $l \geq 1$ and that factor tree T_v^l of factor graph G exists. Then in iteration l of DTP,*

$$t_v^{(l)} = \mathtt{F}_{T_v^l}^{L[T_v^l]\to v} \left(\prod_{u\in L[T_v^l]} t^{(1)}_{T_v^l \; u\xrightarrow{} v} \right).$$

The implication of this result is that on factor graph with sufficiently large girth, DTP is in fact very well-behaved: the summary token at any variable x_v in iteration l depends precisely on the initial tokens passed by variables that are $2l$ away from x_v. Specifically, one may view those tokens form a rectangle on $L[T_v^l]$, and the summary token at x_v in iteration l is precisely the set of all assignments on $\{v\}$ that can make $\Gamma_{T_v^l}$ satisfied, given the assignment on $L[T_v^l]$ is from that rectangle.

Now we develop some results of DTP that require no "local cycle-freeness" in the factor graph.

Lemma 10 *At every $v \in V$ and for any l,*

$$t_v^{(l)} = \bigcap_{c\in C(v)} t_{v\to c}^{(l+1)}.$$

Proof: Suppose that $x_v \in t_v^{(l)}$. Then $x_v \in t_{c\to v}^{(l)}$ for every $c \in C(v)$, by the definition of summary messages. It follows that $x_v \in t_{v\to c}^{(l+1)}$ for every $c \in C(v)$. Then $x_v \in \bigcap_{c\in C(v)} t_{v\to c}^{(l+1)}$. This shows that $t_v^{(l)} \subseteq \bigcap_{c\in C(v)} t_{v\to c}^{(l+1)}$.

On the other hand, suppose that $x_v \in \bigcap_{c\in C(v)} t_{v\to c}^{(l+1)}$. Then $x_v \in t_{v\to c}^{(l+1)} = \bigcap_{b\in C(v)\setminus\{c\}} t_{b\to v}^{(l)}$, for every $c \in C(v)$. It follows that $x_v \in t_{b\to v}^{(l)}$ for every $b \in C(v)$, giving rise to that $x_v \in \bigcap_{b\in C(v)} t_{b\to v}^{(l)} = t_v^{(l)}$. Thus $\bigcap_{c\in C(v)} t_{v\to c}^{(l+1)} \subseteq t_v^{(l)}$.

Therefore $t_v^{(l)} = \bigcap_{c\in C(v)} t_{v\to c}^{(l+1)}$. ■

Lemma 11 *Suppose that $\hat{x}_V$ is a satisfying assignment on V, namely that $\hat{x}_V$ satisfies (3.1). If $\hat{x}_V \in \prod_{v\in V} \bigcap_{c\in C(v)} t_{v\to c}^{(l)}$ in some iteration l, then $\hat{x}_V \in \prod_{v\in V} t_v^{(l)}$.*

Proof: The fact that $\hat{x}_V \in \prod_{v\in V} \bigcap_{c\in C(v)} t_{v\to c}^{(l)}$ implies that for every $v \in V$ and $c \in C(v)$, $\hat{x}_{V:\{v\}} \in \bigcap_{c\in C(v)} t_{v\to c}^{(l)} \subseteq t_{v\to c}^{(l)}$, and hence via the "monotonicity" of function $\mathtt{F}_c$, $\mathtt{F}_c\left(\{\hat{x}_{V:V(c)\setminus\{v\}}\}\right) \subseteq \mathtt{F}_c\left(\prod_{u\in V(c)\setminus\{v\}} t_{u\to c}^{(l)}\right) = t_{c\to v}^{(l)}$. Incorporating that $\hat{x}_V$ is a satisfying assignment, we see that $\hat{x}_{V:\{v\}} \in \mathtt{F}_c\left(\{\hat{x}_{V:V(c)\setminus\{v\}}\}\right) \subseteq t_{c\to v}^{(l)}$, for every $v \in V$ and $c \in C(v)$. Thus $\hat{x}_{V:\{v\}} \in \bigcap_{c\in C(v)} t_{c\to v}^{(l)} = t_v^{(l)}$. It then follows that $\hat{x}_V \in \prod_{v\in V} t_v^{(l)}$. ■

Proposition 2 *Suppose that $\hat{x}_V$ is a satisfying assignment and that the initialization of DTP is such that $\hat{x}_{V:\{v\}} \in t_{v\to c}^{(1)}$ for every $v \in V$ and $c \in C(v)$. Then in any iteration l, the rectangle $\prod_{v\in V} t_v^{(l)}$ formed by the summary tokens contains $\hat{x}_V$.*

Proof: At iteration 1, the fact that $\hat{x}_{V:\{v\}} \in t_{v\to c}^{(1)}$ for every $v \in V$ and $c \in C(v)$ implies that $\hat{x}_V \in \prod_{v\in V} \bigcap_{c\in C(v)} t_{v\to c}^{(1)}$. Followed by Lemma 11, we have $\hat{x}_V \in \prod_{v\in V} t_v^{(1)}$.

As the inductive hypothesis, suppose we have $\hat{x}_V \in \prod_{v\in V} t_v^{(l)}$ at iteration l. At iteration $l+1$, followed by Lemma 10, we have $\hat{x}_V \in \prod_{v\in V} \bigcap_{c\in C(v)} t_{v\to c}^{(l+1)}$. Then by Lemma 11, $\hat{x}_V \in \prod_{v\in V} t_v^{(t+1)}$.

Therefore, this proposition is proved by induction. ■

At this end, we have shown that if DTP is initialized to "containing" a satisfying assignment, then this assignment is contained in the rectangle formed by the summary tokens in all iterations. That is, the solution of the CSP will never get "lost" during DTP iteration provided that it is contained in the initial rectangle. This result (Proposition 2) and Corollary 1 presented earlier will become useful when we discuss the dynamics of PTP.

4.5.2 On the Dynamics of PTP and Weighted PTP

We now turn our attention to (non-weighted) PTP.

Denote by G_v^l the factor-subgraph of G which contains all factors whose messages have propagated to variable x_v by the end of PTP iteration l. That is, G_v^l is the factor-subgraph of G that contains variable vertex x_v and all vertices whose distances to x_v are no larger than $2l$. It is apparent that if G_v^l is a tree, then it is the (v, l) factor tree T_v^l.

Let l^* be the smallest l such that at least for one $v \in V$, T_v^l does not exist. Denote $m_v(l) := \left|\left(\Gamma_{G_v^l}\right)_{:\{v\}}\right|$. That is, $m_v(l)$ is the number of assignments of variable x_v that can make all constraints in G_v^l satisfied. Clearly, $m_v(l)$ is a non-increasing function of l.

We will first restrict the CSP to a "single-solution CSP", i.e., having exactly one satisfying assignment. We will denote this assignment on V by $\hat{x}_V$.

Let $\hat{l}$ be the smallest l for which $\min_v m_v(l) = 1$. It is worth noting that such $\hat{l}$ exists since the CSP has precisely one solution. Let $\hat{v}$ satisfy $m_{\hat{v}}(\hat{l}) = 1$.

Proposition 3 *Let factor graph G represent a single-solution CSP. Suppose that the initialization of PTP is such that every left message $\lambda_{v\to c}^{(1)}(t)$ is strictly positive for every $t \in (\chi^*)^{\{v\}}$. If $\hat{l} < l^*$, then*

$$\mu_{\hat{v}}^{\text{norm }(\hat{l})}(t) = [t = \{\hat{x}_{V:\{\hat{v}\}}\}].$$

Proof: This result relies on Corollary 1.

First, $\hat{l} < l^*$ implies that $(\hat{v}, \hat{l})$ factor tree $T_{\hat{v}}^{\hat{l}}$ exists. Then by Corollary 1, if DTP is initialized such that the tokens sent from the leaves of $T_{\hat{v}}^{\hat{l}}$ form $\prod_{u \in L[T_{\hat{v}}^{\hat{l}}]} t^{(1)}_{u \xrightarrow{T_{\hat{v}}^{\hat{l}}} \hat{v}}$, then the summary token at

v in the $\hat{l}^{\text{th}}$ iteration is $\mathtt{F}_{T_{\hat{v}}^{\hat{l}}}^{L[T_{\hat{v}}^{\hat{l}}]\to\hat{v}}\left(\prod_{u\in L[T_{\hat{v}}^{\hat{l}}]\ \ u\xrightarrow{T_{\hat{v}}^{\hat{l}}}\hat{v}} t^{(1)}\right)$.

Since $\hat{v}$ satisfies $m_{\hat{v}}(\hat{l}) = 1$, it is necessary that $\mathtt{F}_{T_{\hat{v}}^{\hat{l}}}^{L[T_{\hat{v}}^{\hat{l}}]\to\hat{v}}\left(\prod_{u\in L[T_{\hat{v}}^{\hat{l}}]\ \ u\xrightarrow{T_{\hat{v}}^{\hat{l}}}\hat{v}} t^{(1)}\right)$ is either token $\{\hat{x}_{V:\{v\}}\}$ or $\emptyset$, which depends on the rectangle initialized.

Now PTP on $T_{\hat{v}}^{\hat{l}}$, with respect to $x_{\hat{v}}$, may be understood as initializing a *random* rectangle on $L[T_{\hat{v}}^{\hat{l}}]$ (the distribution of which is characterized by the product of the initial messages), transforming the random rectangle to random token on $\hat{v}$ via a functional mapping $\mathtt{F}_{T_{\hat{v}}^{\hat{l}}}^{L[T_{\hat{v}}^{\hat{l}}]\to\hat{v}}(\cdot)$, and conditioning on the resulting token being valid (non-empty set). The fact that initial messages of PTP are strictly positive assures that every rectangle on $L[T_{\hat{v}}^{\hat{l}}]$ has non-zero probability during initialization. After conditioning on the resulting token being valid, the token $\emptyset$ is removed from the allowed realization of the resulting token and thus the resulting token equals $\{\hat{x}_{V:\{v\}}\}$ with probability 1. This completes the proof. ■

This result and its proof can be easily extended to a somewhat larger family of CSPs each containing multiple solutions, as shown in the next proposition.

Proposition 4 *Suppose that in the CSP, there exists a coordinate $\hat{v} \in V$ and an assignment $\hat{x}_{\hat{v}} \in (\chi^*)^{\{v\}}$ such that every satisfying configuration $\tilde{x}_V \in \Gamma$ satisfies $\tilde{x}_{V:\{v\}} = \hat{x}_v$. If for some integer $\hat{l}$, $T_{\hat{v}}^{\hat{l}}$ exists and $m_{\hat{v}}(\hat{l}) = 1$, then*

$$\mu_{\hat{v}}^{\text{norm }(\hat{l})}(t) = [t = \{\hat{x}_{\hat{v}}\}].$$

The proof is similar to that for proposition 3, which essentially relies on Corollary 1 and that the local tree rooted at $\hat{v}$ is large enough. Skipping the proof, we note that Proposition 3 may be viewed as a special case of Proposition 4.

Based on the results above, we provide some remarks concerning the dynamics of PTP and argue intuitively how it solves a CSP.

1. Similar to what was argued in the proof of Proposition 3, the key insight regarding

what PTP is doing is that PTP updates a *random* rectangle whose sides are distributed independently.

At the initialization stage, PTP defines a random rectangle on V, where the sides of the random rectangles are treated as independent random variables. In every iteration, PTP maps this random rectangle to a new random rectangle in the following steps.

(a) Apply a functional mapping defined by the right-message update rule and the left-message update rule.

(b) Eliminate the resulting empty rectangles (via conditioning on that each side of the resulting random rectangle is not the empty set and re-normalization).

(c) Take the marginal distribution of the resulting random rectangle on each side variable, and treat all sides as being independent random variables. This defines a new random rectangle.

PTP iterates over these steps to continuously update the random rectangle.

2. For single-solution CSPs, based on Proposition 3, if the girth of the graph is large enough, at least one side of the new rectangle, after some iterations, becomes deterministic, namely the singleton set containing the correct assignment for that variable. This would allow the decimation procedure to fix this variable to the correct assignment and reduce the problem. Similar results hold for CSPs having more than one solutions but in which all solutions share a single assignment on some coordinate. By Proposition 4, in this case, when the local tree rooted at that variable is sufficiently large, PTP will find that variable and its correct assignment. Of course, the condition of Proposition 3 and that of Proposition 4, namely that there is a sufficiently large local tree rooted at a variable and that the variable only has one correct assignment, may not hold in reality. As a consequence, no side of the random rectangle is deterministically a singleton. In that case, the decimation procedure must deal with this ambiguity — resulted from non-ideal

factor graph structure and the complexity of the solution space — and make a good guess to fix a variable.

3. Proposition 3 and Proposition 1 also suggest that when the graph has large girth (and when the solutions share one common assignment on some coordinate), as PTP iterates, the rectangles containing no solutions will be gradually removed from the sample space of the random rectangle.

4. Proposition 2 implies that regardless of cycle structure of the graph, all solution-containing rectangles will be kept (possibly in a form of combining each other) over PTP iterations.

5. Combining 3) and 4) above, one may view each PTP iteration as performing a "filtering" operation on the distribution of the random rectangle. As the distribution of the random rectangle evolves, the probability mass moves gradually to one biased to some solution-containing rectangles. When the graph has large girth and some coordinate is in a "favorable" position (in a sense combining its location in the graph and its role in the solution space), the summary message at this coordinate may become more deterministically biased to a singleton token, making decimation possible.

Finally, we briefly remark on weighted PTP.

Similar to PTP, weighted PTP also updates a random rectangle. However, instead of using a functional mapping, in step a) of the above procedure, it uses a conditional distribution. By examining the form of the obedience conditionals, it is intuitive that comparing with PTP, weighted PTP shifts the distribution of each side of the random rectangle more towards "smaller" tokens on each coordinate. (Here t_v is said to be smaller than t'_v if $t_v \subset t'_v$.) This provides the algorithm better opportunity to lead to some side of the random rectangle more deterministically biased to a singleton.

4.6 Concluding Remarks

Previous generalizations of SP have been either from the perspective of extending SP to non-binary alphabet, or from the perspective of introducing "weights" to SP message-update equations. In this chapter, we consider a full generalization of SP, taking into account both perspectives.

By an appropriate extension of the variable alphabet to a set of "tokens", we show that SP (on any alphabet) may be regarded as a "probabilistic token passing" procedure, by which random tokens are passed by the vertices of the factor graph representing the CSP. When each vertex in the factor graph, generates outgoing tokens via a probabilistic dependency of the incoming tokens, PTP can be generalized to weighted PTP. We show that weighted PTP generalizes previously reported SP algorithms for k-SAT problems and for q-COL problems.

We show that PTP and weighted PTP may be viewed as iteratively updating a random "rectangle". Under certain conditions, we provide analytic results concerning how such updates lead to the correct solution of the CSP. These results are however still preliminary. Further investigation is required to understand the dynamics of PTP and weighted PTP.

Chapter 5

Connection To Belief Propagation

At this point, we have identified SP with an equivalent but probabilistically interpretable algorithmic procedure, PTP, and generalized weighted SP from the special case of k-SAT and binary problems to arbitrary CSPs, in terms of weighted PTP. Now we are in the position to discuss the reduction of SP from BP, where we will refer to SP exclusively as PTP, and weighted SP exclusively as weighted PTP.

As is well known, the derivation of the BP algorithm is based on a well-defined factoring function, or seen from a probabilistic perspective, a Markov random field (MRF). Thus, whether PTP or weighted PTP may be reduced from BP boils down to whether there is an MRF formulation on which the derived BP algorithm coincides with PTP or weighted PTP. In [23], an MRF is constructed for k-SAT problem, on which BP reduces to what we now call weighted PTP.

In this chapter, we first generalize the MRF formalism, in the style of [23], to arbitrary CSPs, and derive the corresponding BP algorithm. Our formalism uses Forney graphs where we explicitly introduce state variables to make correspondence to the left and right messages in SP. We then investigate whether the derived BP algorithm may be reduced to PTP or weighted PTP. We will begin this investigation with the special case of k-SAT problems, and then proceed to the 3-COL problems and to general CSPs. For k-SAT problems, we show

that the BP algorithm on the normally realized MRF is readily reducible to weighted PTP as long as the BP messages are initialized to satisfy certain condition. This reduction is cleaner and more transparent comparing with the SP-to-BP reduction presented in [23]. We note that the initialization condition, when satisfied in the first BP iteration, will necessarily be satisfied in later iterations in k-SAT problems. Identifying the important role of this condition, we call this condition the *state-decoupling condition*. However, as we proceed to show, in 3-COL problems, it is impossible for the state-decoupling condition to hold true non-trivially across all BP iterations. Nevertheless, if one manually manipulate the BP messages to impose this condition in every iteration, which results in a modified BP message-update rule referred to as *state-decoupled BP* or SDBP in short, then the (SD)BP messages will still reduce to PTP. This on one hand justifies the role of the state-decoupling condition in BP-to-PTP reduction, and on the other hand suggests that for general CSPs, PTP (or SP) is not a special case of the BP algorithm. We then proceed further by investigating whether the state-decoupling condition is sufficient for BP to reduce to PTP or weighted PTP for general CSPs. To that end, we show that yet another "local compatibility" condition concerning the structure of the CSP (in terms of the interaction between neighboring constraints) is required for SDBP to reduce to PTP or weighted PTP.

5.1 Normally Realized Markov Random Field

Given a CSP represented by factor graph G, we now define its corresponding *normally realized Markov random field* $\tilde{G}$ using a Forney graph representation [17]. We note that random variables involved in the probability mass function (PMF) represented by $\tilde{G}$ are no longer those associated with factor graph (or equivalently MRF) G, but rather a new set of random variables, each distributed over the set of *tokens* associated with a coordinate. Additionally, as the central component of the Forney graph, another set of random variables, typically called *generalized states* or simply *states*, are also included.

Specifically, as a graph, $\tilde{G}$ can be constructed by adding a "half-edge" to each variable vertex of G. We now define each variable and local function in $\tilde{G}$.

- Each local function (or vertex) in $\tilde{G}$ corresponding to variable vertex x_v in G will be denoted by $g_v(\cdot)$, and referred to as a *left function*.
- Each local function (or vertex) in $\tilde{G}$ corresponding to function vertex Γ_c will be denoted by $f_c(\cdot)$, and referred to as a *right function*.
- The half edge incident on g_v represents variable y_v, referred to as a *side*, taking values from $(\chi^*)^{\{v\}}$.
- The edge connecting left function g_v and right function f_c represents variable $s_{v,c}$, referred to as a *state*, taking values from $(\chi^*)^{\{v\}} \times (\chi^*)^{\{v\}}$. We will also write state $s_{v,c}$ as pair $\left(s^L_{v,c}, s^R_{v,c}\right)$ of *left state* $s^L_{v,c}$ and *right state* $s^R_{v,c}$.
- Left function g_v for $v \in V$ is defined as

$$g_v(y_v, s_{v,C(v)}) := \omega_v \left(y_v \left| \bigcap_{c \in C(v)} s^R_{v,c} \right. \right) \cdot \prod_{c \in C(v)} [s^L_{v,c} = y_v], \tag{5.1}$$

  where $s_{v,C(v)}$ is the short-hand notation for $\langle s_{v,c} \rangle_{c \in C(v)}$ and ω_v is an obedience conditional on $(\chi^*)^{\{v\}}$.

- Right function f_c for each $c \in C$ is defined as

$$f_c(s_{V(c),c}) := \prod_{v \in V(c)} [s^R_{v,c} = \mathtt{F}_c(s^L_{V(c)\setminus\{v\},c})], \tag{5.2}$$

  where $s_{V(c),c}$ is the short-hand notation for $\langle s_{v,c} \rangle_{v \in V(c)}$.

- The global function represented by $\tilde{G}$ is

$$F(y_V, s_{V,C}) := \prod_{v \in V} g_v(y_v, s_{v,C(v)}) \cdot \prod_{c \in C} f_c(s_{V(c),c}), \tag{5.3}$$

where $s_{V,C}$ is the short-hand notation for $\{s_{v,c} : \forall(v-c)\}$.

It is clear that upon normalization, function F may represent a PMF and the factorization of F encoded by $\tilde{G}$ realizes an MRF. An example of such normally realized MRF, corresponding to the toy 3-SAT problem in Figure 3.1, is given in Figure 5.1.

Using the "intention-command" analogy, one may view that for any v, both y_v and each left state $s^L_{v,c}$ stores the intention of variable x_v, and that for any given c, each right state $s^R_{v,c}$ stores the command of constraint Γ_c sent to variable x_v. The intention of variable x_v depends on the intersection of all incoming commands probabilistically via the obedience conditional ω_v. The command of Γ_c sent to each variable x_v need to equal the forced token by the rectangle formed by the intentions from all other neighboring variables.

Normally realized MRF $\tilde{G}$ as defined in (5.3) may be interpreted as a distribution of rectangles. Specifically, we say a rectangle y_V is *valid* under F if there exists a configuration of $s_{V,C}$ such that $(y_V, s_{V,C})$ is valid under F. Then it immediately follows that the PMF represented by MRF $\tilde{G}$, upon marginalizing over states $s_{V,C}$, characterizes the set of all valid rectangles under F (via the support of the marginal of F on y_V).

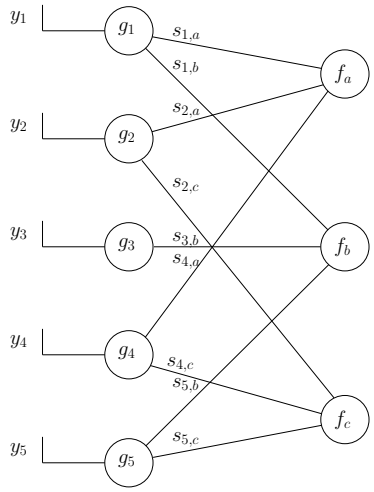


Figure 5.1: The Forney graph representing the normal realization of the toy problem in Figure 3.1.

A simple property of such MRFs is given in the following lemma, which immediately follows

from the definition of the left functions.

Lemma 12 *If configuration $(y_V, s_{V,C})$ is valid under F, then it holds for every $(v-c)$ that*

$$s^L_{v,c} = y_v \subseteq s^R_{v,c}.$$

Now we consider applying the BP message-update rule on the Forney graph $\tilde{G}$ we just defined, where we will use $\rho_{c\to v}$ (referred to as a right message) and $\lambda_{v\to c}$ (referred to as a left message) to denote the message passed from a right function f_c to a left function g_v and the message passed from left function g_v to right function f_c respectively, and use μ_v to denote the summary message at variable y_v. We note that both right message $\rho_{c\to v}$ and left message $\lambda_{v\to c}$ are functions on the state space $(\chi^*)^{\{v\}} \times (\chi^*)^{\{v\}}$.

Lemma 13 *The BP message-update rule on Forney graph $\tilde{G}$ is:*

$$\lambda_{v\to c}(s^L_{v,c}, s^R_{v,c}) := \sum_{s^R_{v,C(v)\setminus\{c\}}} \omega_v\left(s^L_{v,c}\middle|\bigcap_{b\in C(v)} s^R_{v,b}\right) \prod_{b\in C(v)\setminus\{c\}} \rho_{b\to v}(s^L_{v,c}, s^R_{v,b}) \tag{5.4}$$

$$\rho_{c\to v}(s^L_{v,c}, s^R_{v,c}) := \sum_{s^L_{V(c)\setminus\{v\},c}} \left[s^R_{v,c} = \mathtt{F}_c(s^L_{V(c)\setminus\{v\},c})\right] \prod_{u\in V(c)\setminus\{v\}} \lambda_{u\to c}(s^L_{u,c}, \mathtt{F}_c(s^L_{V(c)\setminus\{u\},c})) \tag{5.5}$$

$$\mu_v(y_v) := \sum_{s^R_{v,C(v)}} \omega_v\left(y_v\middle|\bigcap_{c\in C(v)} s^R_{v,c}\right) \prod_{c\in C(v)} \rho_{c\to v}(y_v, s^R_{v,c}). \tag{5.6}$$

Before proving this lemma, it is useful to note the following elementary results.

Lemma 14 *1. For any function ϕ,*

$$\sum_y \phi(x,y)[y=z] = \phi(x,z). \tag{5.7}$$

2. For any collection of functions $\phi_1, \phi_2, \ldots, \phi_m$,

$$\sum_{x_1,x_2,\ldots,x_n} \prod_{i=1}^n \phi_i(x_i) = \prod_{i=1}^n \sum_{x_i} \phi_i(x_i). \tag{5.8}$$

We now prove Lemma 13.

Proof:

$$\begin{aligned}
\lambda_{v\to c}(s^L_{v,c}, s^R_{v,c}) &= \sum_{y_v} \sum_{s_{v,C(v)\setminus\{c\}}} \omega_v\left(y_v \middle| \bigcap_{b\in C(v)} s^R_{v,b}\right) \prod_{b\in C(v)} [s^L_{v,b} = y_v] \prod_{b\in C(v)\setminus\{c\}} \rho_{b\to v}(s^L_{v,b}, s^R_{v,b}) \\
&= \sum_{y_v} [s^L_{v,c} = y_v] \sum_{s^R_{v,C(v)\setminus\{c\}}} \omega_v\left(y_v \middle| \bigcap_{b\in C(v)} s^R_{v,b}\right) \sum_{s^L_{v,C(v)\setminus\{c\}}} \prod_{b\in C(v)\setminus\{c\}} \left(\rho_{b\to v}(s^L_{v,b}, s^R_{v,b}) \cdot [s^L_{v,b} = y_v]\right) \\
&\overset{(5.8)}{=} \sum_{y_v} [s^L_{v,c} = y_v] \sum_{s^R_{v,C(v)\setminus\{c\}}} \omega_v\left(y_v \middle| \bigcap_{b\in C(v)} s^R_{v,b}\right) \prod_{b\in C(v)\setminus\{c\}} \sum_{s^L_{v,b}} \left(\rho_{b\to v}(s^L_{v,b}, s^R_{v,b}) \cdot [s^L_{v,b} = y_v]\right) \\
&\overset{(5.7)}{=} \sum_{y_v} [s^L_{v,c} = y_v] \sum_{s^R_{v,C(v)\setminus\{c\}}} \omega_v\left(y_v \middle| \bigcap_{b\in C(v)} s^R_{v,b}\right) \prod_{b\in C(v)\setminus\{c\}} \rho_{b\to v}(y_v, s^R_{v,b}) \\
&= \sum_{s^R_{v,C(v)\setminus\{c\}}} \omega_v\left(s^L_{v,c} \middle| \bigcap_{b\in C(v)} s^R_{v,b}\right) \prod_{b\in C(v)\setminus\{c\}} \rho_{b\to v}(s^L_{v,c}, s^R_{v,b}).
\end{aligned}$$

$$\begin{aligned}
\rho_{c\to v}(s^L_{v,c}, s^R_{v,c}) &= \sum_{s_{V(c)\setminus\{v\},c}} \prod_{u\in V(c)} [s^R_{u,c} = \mathtt{F}_c(s^L_{V(c)\setminus\{u\},c})] \prod_{u\in V(c)\setminus\{v\}} \lambda_{u\to c}(s^L_{u,c}, s^R_{u,c}) \\
&= \sum_{s^L_{V(c)\setminus\{v\},c}} [s^R_{v,c} = \mathtt{F}_c(s^L_{V(c)\setminus\{v\},c})] \sum_{s^R_{V(c)\setminus\{v\},c}} \prod_{u\in V(c)\setminus\{v\}} \left([s^R_{u,c} = \mathtt{F}_c(s^L_{V(c)\setminus\{u\},c})] \cdot \lambda_{u\to c}(s^L_{u,c}, s^R_{u,c})\right) \\
&\overset{(5.8)}{=} \sum_{s^L_{V(c)\setminus\{v\},c}} [s^R_{v,c} = \mathtt{F}_c(s^L_{V(c)\setminus\{v\},c})] \prod_{u\in V(c)\setminus\{v\}} \sum_{s^R_{u,c}} \left([s^R_{u,c} = \mathtt{F}_c(s^L_{V(c)\setminus\{u\},c})] \cdot \lambda_{u\to c}(s^L_{u,c}, s^R_{u,c})\right) \\
&\overset{(5.7)}{=} \sum_{s^L_{V(c)\setminus\{v\},c}} [s^R_{v,c} = \mathtt{F}_c(s^L_{V(c)\setminus\{v\},c})] \prod_{u\in V(c)\setminus\{v\}} \lambda_{u\to c}(s^L_{u,c}, \mathtt{F}_c(s^L_{V(c)\setminus\{u\},c})).
\end{aligned}$$

$$\begin{aligned}
\mu_v(y_v) &= \sum_{s_{v,C(v)}} \omega_v\left(y_v \middle| \bigcap_{c\in C(v)} s^R_{v,c}\right) \prod_{c\in C(v)} [s^L_{v,c} = y_v] \prod_{c\in C(v)} \rho_{c\to v}(s^L_{v,c}, s^R_{v,c}) \\
&= \sum_{s^R_{v,C(v)}} \omega_v\left(y_v \middle| \bigcap_{c\in C(v)} s^R_{v,c}\right) \sum_{s^L_{v,C(v)}} \prod_{c\in C(v)} \left([s^L_{v,c} = y_v] \cdot \rho_{c\to v}(s^L_{v,c}, s^R_{v,c})\right) \\
&\overset{(5.8)}{=} \sum_{s^R_{v,C(v)}} \omega_v\left(y_v \middle| \bigcap_{c\in C(v)} s^R_{v,c}\right) \prod_{c\in C(v)} \sum_{s^L_{v,c}} \left([s^L_{v,c} = y_v] \cdot \rho_{c\to v}(s^L_{v,c}, s^R_{v,c})\right) \\
&\overset{(5.7)}{=} \sum_{s^R_{v,C(v)}} \omega_v\left(y_v \middle| \bigcap_{c\in C(v)} s^R_{v,c}\right) \prod_{c\in C(v)} \rho_{c\to v}(y_v, s^R_{v,c}).
\end{aligned}$$

■

5.2 Weighted PTP as BP for k-SAT

Now we show that for k-SAT problems, weighted PTP *is* an instance of BP when the parametrization of weighted PTP is consistent with the parametrization of the normally realized MRF from which BP is derived.

We begin with introducing a simplification of notations. For any $(v - c)$ and edge label $L_{v,c}$, we will write $\mathbf{L_{v,c}}$ as $\mathbf{L}$, and $\bar{\mathbf{L}}_{\mathbf{v,c}}$ as $\bar{\mathbf{L}}$. This suppression of the dependency of $\mathbf{L_{v,c}}$ and $\bar{\mathbf{L}}_{\mathbf{v,c}}$ on their subscripts should not result in any ambiguity, when the context clearly indicates the subscript (v, c) or the edge to which the edge label $L_{v,c}$ refers. Additionally, for any $v \in V$, we will write $\mathbf{01}_v$ as $*$. Thus, each left or right state will take configurations from set $\{\mathbf{L}, \bar{\mathbf{L}}, *\}$, where the interpretation of $\mathbf{L}$ and $\bar{\mathbf{L}}$ depends on the edge with which the state is associated. For any given configuration of a state $(s^L_{v,c}, s^R_{v,c})$, we will suppress the comma between the left-state configuration and the right-state configuration. For example, state configurations $(\mathbf{L}, *)$, $(\bar{\mathbf{L}}, *)$, $(*, *)$ and $(\mathbf{L}, \mathbf{L})$ will be written respectively as $\mathbf{L}*, \bar{\mathbf{L}}*, **$ and $\mathbf{LL}$.

Lemma 15 *Let F be defined via (5.1), (5.2) and (5.3), where each weighting function ω_v is defined in (4.39). If $(y_V, s_{V,C})$ is valid under F, then*

1. *for every $(v - c)$, it holds that $s^R_{v,c} \neq \bar{\mathbf{L}}$, $s_{v,c} \neq \bar{\mathbf{L}}\mathbf{L}$ and that $s_{v,c} \neq *\mathbf{L}$, and*

2. *$F(y_V, s_{V,C}) = \epsilon^{n_{*|*}(y_V, s_{V,C})} \cdot (1 - \epsilon)^{n_{\cdot|*}(y_V, s_{V,C})}$, where $n_{*|*}(y_V, s_{V,C})$ and $n_{\cdot|*}(y_V, s_{V,C})$ are respectively the cardinalities of set $\{v \in V : y_v = \bigcap_{c \in C(v)} s^R_{v,c} = *\}$ and set $\{v \in V : y_v \subset \bigcap_{c \in C(v)} s^R_{v,c} = *\}$.*

Proof: For part 1, first we observe that $s^R_{v,c} \neq \bar{\mathbf{L}}$, directly following from the definition of the right function (5.2). Then by Lemma 12, it is easy to see that $s_{v,c} \neq \bar{\mathbf{L}}\mathbf{L}$ and that $s_{v,c} \neq *\mathbf{L}$.

For part 2, we may proceed as follows.

$$
\begin{aligned}
F(y_V, s_{V,C}) &= \prod_{v\in V} g_v(y_v, s_{v,C(v)}) \cdot \prod_{c\in C} f_c(s_{V(c),c}) \\
&\overset{(5.1),(5.2)}{=} \prod_{v\in V}\left(\omega_v\left(y_v \,\middle|\, \bigcap_{c\in C(v)} s^R_{v,c}\right)\cdot \prod_{c\in C(v)} [s^L_{v,c} = y_v]\right)\cdot \prod_{c\in C}\prod_{v\in V(c)} [s^R_{v,c} = \mathtt{F}_c(s^L_{V(c)\setminus\{v\},c})] \\
&\overset{(a)}{=} \prod_{v\in V}\omega_v\left(y_v \,\middle|\, \bigcap_{c\in C(v)} s^R_{v,c}\right) \\
&\overset{(b)}{=} \epsilon^{n_{*|*}(y_V, s_{V,C})}\cdot(1-\epsilon)^{n_{\cdot|*}(y_V, s_{V,C})},
\end{aligned}
$$

where equality (a) is due to the fact that $(y_V, s_{V,C})$ is valid under F, and equality (b) follows from the definition of the weighting function $\omega\left(y_v \left| \bigcap_{c\in C(v)} s^R_{v,c}\right.\right)$ in (4.39). ■

The second part of this lemma, as a slight digression, suggests that the PMF under this MRF model is identical to that of [23], since an equivalent result is shown for the MRF in [23]. We note that the MRF in [23] serves as a combinatorial framework for the study of k-SAT problems, which leads to further insights of SP for k-SAT problems (the reader is referred to [23] for additional results). To a certain extent, one may expect that the normally realized MRF presented here may serve similar purposes for general CSPs.

The first part of this lemma suggests that although each state takes on values from $\{\mathbf{L}, \bar{\mathbf{L}}, *\}\times \{\mathbf{L}, \bar{\mathbf{L}}, *\}$, there are in fact only four possible state configurations that contribute to defining a valid rectangle. When applying the BP message-update rule on the Forney graph representation of a k-SAT problem, this implies that messages $\lambda_{v\to c}$, $\rho_{c\to v}$ and μ_v are all supported by $\{\mathbf{LL}, \bar{\mathbf{L}}*, \mathbf{L}*, **\}$.

The BP message-update rule is given in Lemma 16, which directly follows from equations (5.4) to (5.6).

Lemma 16 *The BP message-update rule applied on Forney graph $\tilde{G}$ of a k-SAT problem gives*

rise to:

$$\lambda_{v\to c}(\mathbf{LL}) := \prod_{b\in C_c^{\mathrm{u}}(v)} \rho_{b\to v}(\bar{\mathbf{L}}*) \prod_{b\in C_c^{\mathrm{s}}(v)} (\rho_{b\to v}(\mathbf{LL}) + \rho_{b\to v}(\mathbf{L}*)) \tag{5.9}$$

$$\lambda_{v\to c}(\bar{\mathbf{L}}*) := \prod_{b\in C_c^{\mathrm{s}}(v)} \rho_{b\to v}(\bar{\mathbf{L}}*) \left(\prod_{b\in C_c^{\mathrm{u}}(v)} (\rho_{b\to v}(\mathbf{L}*) + \rho_{b\to v}(\mathbf{LL})) - \epsilon \prod_{b\in C_c^{\mathrm{u}}(v)} \rho_{b\to v}(\mathbf{L}*) \right) \tag{5.10}$$

$$\lambda_{v\to c}(\mathbf{L}*) := \prod_{b\in C_c^{\mathrm{u}}(v)} \rho_{b\to v}(\bar{\mathbf{L}}*) \left(\prod_{b\in C_c^{\mathrm{s}}(v)} (\rho_{b\to v}(\mathbf{L}*) + \rho_{b\to v}(\mathbf{LL})) - \epsilon \prod_{b\in C_c^{\mathrm{s}}(v)} \rho_{b\to v}(\mathbf{L}*) \right) \tag{5.11}$$

$$\lambda_{v\to c}(**) :=\epsilon \prod_{b\in C_c^{\mathrm{u}}(v)\cup C_c^{\mathrm{s}}(v)} \rho_{b\to v}(**) \tag{5.12}$$

$$\rho_{c\to v}(\mathbf{LL}) := \prod_{u\in V(c)\setminus\{v\}} \lambda_{u\to c}(\bar{\mathbf{L}}*) \tag{5.13}$$

$$\begin{aligned}\rho_{c\to v}(\bar{\mathbf{L}}*) := & \prod_{u\in V(c)\setminus\{v\}} (\lambda_{u\to c}(\mathbf{L}*) + \lambda_{u\to c}(**) + \lambda_{u\to c}(\bar{\mathbf{L}}*)) \\ & + \sum_{u\in V(c)\setminus\{v\}} (\lambda_{u\to c}(\mathbf{LL}) - \lambda_{u\to c}(\mathbf{L}*) - \lambda_{u\to c}(**)) \prod_{w\in V(c)\setminus\{u,v\}} \lambda_{w\to c}(\bar{\mathbf{L}}*) \\ & - \prod_{u\in V(c)\setminus\{v\}} \lambda_{u\to c}(\bar{\mathbf{L}}*)\end{aligned} \tag{5.14}$$

$$\rho_{c\to v}(\mathbf{L}*) := \prod_{u\in V(c)\setminus\{v\}} (\lambda_{u\to c}(\mathbf{L}*) + \lambda_{u\to c}(**) + \lambda_{u\to c}(\bar{\mathbf{L}}*)) - \prod_{u\in V(c)\setminus\{v\}} \lambda_{u\to c}(\bar{\mathbf{L}}*) \tag{5.15}$$

$$\rho_{c\to v}(**) := \prod_{u\in V(c)\setminus\{v\}} (\lambda_{u\to c}(\mathbf{L}*) + \lambda_{u\to c}(**) + \lambda_{u\to c}(\bar{\mathbf{L}}*)) - \prod_{u\in V(c)\setminus\{v\}} \lambda_{u\to c}(\bar{\mathbf{L}}*) \tag{5.16}$$

$$\mu_v(\mathbf{0}) := \prod_{c\in C^1(v)} \rho_{c\to v}(\bar{\mathbf{L}}*) \left(\prod_{c\in C^0(v)} (\rho_{c\to v}(\mathbf{LL}) + \rho_{c\to v}(\mathbf{L}*)) - \epsilon \prod_{c\in C^0(v)} \rho_{c\to v}(\mathbf{L}*) \right) \tag{5.17}$$

$$\mu_v(\mathbf{1}) := \prod_{c\in C^0(v)} \rho_{c\to v}(\bar{\mathbf{L}}*) \left(\prod_{c\in C^1(v)} (\rho_{c\to v}(\mathbf{LL}) + \rho_{c\to v}(\mathbf{L}*)) - \epsilon \prod_{c\in C^1(v)} \rho_{c\to v}(\mathbf{L}*) \right) \tag{5.18}$$

$$\mu_v(*) := \epsilon \prod_{c\in C(v)} \rho_{c\to v}(**). \tag{5.19}$$

Now we are ready to investigate how these BP messages may reduced to (weighted) PTP messages. It turns out that the following condition has a special role to play in this reduction.

$$\rho_{c\to v}(\mathbf{L}*) = \rho_{c\to v}(\bar{\mathbf{L}}*) = \rho_{c\to v}(**) \tag{5.20}$$

Proposition 5 *In k-SAT problems, if the BP messages are initialized to satisfy condition (5.20), then this condition is satisfied in every BP iteration.*

Proof: We only need to show that if (5.20) is satisfied during initialization, then it is satisfied in the first iteration after initialization. – In fact, noting that $\rho_{c\to v}(\mathbf{L}*) = \rho_{c\to v}(**)$ necessarily holds in each BP iteration due to (5.15) and (5.16), we only need to prove that $\rho_{c\to v}(\bar{\mathbf{L}}*) = \rho_{c\to v}(\mathbf{L}*)$ holds in the first iteration provided BP messages are initialized to satisfy (5.20).

Under this initialization condition, we have, in the first BP iteration after,

$$\begin{aligned}
\lambda_{v\to c}(\mathbf{L}*) + \lambda_{v\to c}(**) &= \prod_{b\in C_c^{\mathrm{u}}(v)} \rho_{b\to v}(\bar{\mathbf{L}}*) \times \left(\prod_{b\in C_c^{\mathrm{s}}(v)} (\rho_{b\to v}(\mathbf{L}*) + \rho_{b\to v}(\mathbf{LL})) - \epsilon \prod_{b\in C_c^{\mathrm{s}}(v)} \rho_{b\to v}(\mathbf{L}*) \right) \\
&\quad + \epsilon \prod_{b\in C_c^{\mathrm{u}}(v)\cup C_c^{\mathrm{s}}(v)} \rho_{b\to v}(**) \\
&= \prod_{b\in C_c^{\mathrm{u}}(v)} \rho_{b\to v}(\bar{\mathbf{L}}*) \prod_{b\in C_c^{\mathrm{s}}(v)} (\rho_{b\to v}(\mathbf{L}*) + \rho_{b\to v}(\mathbf{LL})) \\
&\quad - \epsilon \prod_{b\in C_c^{\mathrm{u}}(v)} \rho_{b\to v}(\bar{\mathbf{L}}*) \prod_{b\in C_c^{\mathrm{s}}(v)} \rho_{b\to v}(\mathbf{L}*) + \epsilon \prod_{b\in C_c^{\mathrm{u}}(v)\cup C_c^{\mathrm{s}}(v)} \rho_{b\to v}(**) \\
&\overset{(a)}{=} \prod_{b\in C_c^{\mathrm{u}}(v)} \rho_{b\to v}(\bar{\mathbf{L}}*) \prod_{b\in C_c^{\mathrm{s}}(v)} (\rho_{b\to v}(\mathbf{L}*) + \rho_{b\to v}(\mathbf{LL})) \\
&= \lambda_{v\to c}(\mathbf{LL}),
\end{aligned}$$

where equality (a) is due to the initialization condition (5.20).

Then in the subsequent update of the right messages, we have

$$\begin{aligned}
\rho_{c\to v}(\bar{\mathbf{L}}*) &= \prod_{u\in V(c)\setminus\{v\}} (\lambda_{u\to c}(\mathbf{L}*) + \lambda_{u\to c}(**) + \lambda_{u\to c}(\bar{\mathbf{L}}*)) \\
&\quad + \sum_{u\in V(c)\setminus\{v\}} (\lambda_{u\to c}(\mathbf{LL}) - \lambda_{u\to c}(\mathbf{L}*) - \lambda_{u\to c}(**)) \prod_{w\in V(c)\setminus\{u,v\}} \lambda_{w\to c}(\bar{\mathbf{L}}*) \\
&\quad - \prod_{u\in V(c)\setminus\{v\}} \lambda_{u\to c}(\bar{\mathbf{L}}*) \\
&\overset{(b)}{=} \prod_{u\in V(c)\setminus\{v\}} (\lambda_{u\to c}(\mathbf{L}*) + \lambda_{u\to c}(**) + \lambda_{u\to c}(\bar{\mathbf{L}}*)) - \prod_{u\in V(c)\setminus\{v\}} \lambda_{u\to c}(\bar{\mathbf{L}}*) \\
&= \rho_{c\to v}(\mathbf{L}*),
\end{aligned}$$

where equality (b) is due to the above result $\lambda_{v\to c}(\mathbf{LL}) = \lambda_{v\to c}(\mathbf{L}*) + \lambda_{v\to c}(**)$. ■

Theorem 3 *In a k-SAT problem, suppose that the following two conditions are imposed in the BP messages.*

1. *For every $(v-c)$, the BP messages are initialized such that (5.20) is satisfied.*

2. *In each BP iteration, $\lambda_{v\to c}$ is scaled to $\lambda^{\text{norm}}_{v\to c}$ such that $\lambda^{\text{norm}}_{v\to c}(\mathbf{L}*)+\lambda^{\text{norm}}_{v\to c}(\bar{\mathbf{L}}*)+\lambda^{\text{norm}}_{v\to c}(**) = 1$, before it is passed along the edge; that is, $\lambda^{\text{norm}}_{v\to c}(s^L_{v,c}, s^R_{v,c}) := \frac{1}{\sum_{s^L_{v,c}} \lambda_{v\to c}(s^L_{v,c},*)} \cdot \lambda_{v\to c}(s^L_{v,c}, s^R_{v,c})$ for every $(s^L_{v,c}, s^R_{v,c})$ in the support of $\lambda_{v\to c}$ and the right messages are updated based on the normalized left messages, namely,*

$$\rho_{c\to v}(\mathbf{LL}) := \prod_{u\in V(c)\setminus\{v\}} \lambda^{\text{norm}}_{u\to c}(\bar{\mathbf{L}}*) \tag{5.21}$$

$$\begin{aligned}\rho_{c\to v}(\bar{\mathbf{L}}*) := & \prod_{u\in V(c)\setminus\{v\}} (\lambda^{\text{norm}}_{u\to c}(\mathbf{L}*) + \lambda^{\text{norm}}_{u\to c}(**) + \lambda^{\text{norm}}_{u\to c}(\bar{\mathbf{L}}*)) \\ & + \sum_{u\in V(c)\setminus\{v\}} (\lambda^{\text{norm}}_{u\to c}(\mathbf{LL}) - \lambda^{\text{norm}}_{u\to c}(\mathbf{L}*) - \lambda^{\text{norm}}_{u\to c}(**)) \prod_{w\in V(c)\setminus\{u,v\}} \lambda^{\text{norm}}_{w\to c}(\bar{\mathbf{L}}*) \\ & - \prod_{u\in V(c)\setminus\{v\}} \lambda^{\text{norm}}_{u\to c}(\bar{\mathbf{L}}*)\end{aligned} \tag{5.22}$$

$$\rho_{c\to v}(\mathbf{L}*) := \prod_{u\in V(c)\setminus\{v\}} (\lambda^{\text{norm}}_{u\to c}(\mathbf{L}*) + \lambda^{\text{norm}}_{u\to c}(**) + \lambda^{\text{norm}}_{u\to c}(\bar{\mathbf{L}}*)) - \prod_{u\in V(c)\setminus\{v\}} \lambda^{\text{norm}}_{u\to c}(\bar{\mathbf{L}}*) \tag{5.23}$$

$$\rho_{c\to v}(**) := \prod_{u\in V(c)\setminus\{v\}} (\lambda^{\text{norm}}_{u\to c}(\mathbf{L}*) + \lambda^{\text{norm}}_{u\to c}(**) + \lambda^{\text{norm}}_{u\to c}(\bar{\mathbf{L}}*)) - \prod_{u\in V(c)\setminus\{v\}} \lambda^{\text{norm}}_{u\to c}(\bar{\mathbf{L}}*). \tag{5.24}$$

Then the correspondence between BP messages and weighted PTP messages is

$$\lambda^{\text{norm(BP)}}_{v\to c}(\mathbf{L}*) \leftrightarrow [L_{v,c}=0]\cdot\lambda^{\text{norm(PTP)}}_{v\to c}(\mathbf{0}) + [L_{v,c}=1]\cdot\lambda^{\text{norm(PTP)}}_{v\to c}(\mathbf{1}) \tag{5.25}$$

$$\lambda^{\text{norm(BP)}}_{v\to c}(\bar{\mathbf{L}}*) \leftrightarrow [L_{v,c}=0]\cdot\lambda^{\text{norm(PTP)}}_{v\to c}(\mathbf{1}) + [L_{v,c}=1]\cdot\lambda^{\text{norm(PTP)}}_{v\to c}(\mathbf{0}) \tag{5.26}$$

$$\lambda^{\text{norm(BP)}}_{v\to c}(**) \leftrightarrow \lambda^{\text{norm(PTP)}}_{v\to c}(*) \tag{5.27}$$

$$\rho_{c\to v}^{(\mathrm{BP})}(\mathbf{L}*) \quad \leftrightarrow \quad \rho_{c\to v}^{\mathrm{norm}(\mathrm{PTP})}(*) \tag{5.28}$$

$$\rho_{c\to v}^{(\mathrm{BP})}(\mathbf{LL}) \quad \leftrightarrow \quad \rho_{c\to v}^{\mathrm{norm}(\mathrm{PTP})}(\mathbf{0}) + \rho_{c\to v}^{\mathrm{norm}(\mathrm{PTP})}(\mathbf{1}) \tag{5.29}$$

$$\mu_{v}^{(\mathrm{BP})}(\mathbf{0}) \quad \leftrightarrow \quad \mu_{v}^{(\mathrm{PTP})}(\mathbf{0}) \tag{5.30}$$

$$\mu_{v}^{(\mathrm{BP})}(\mathbf{1}) \quad \leftrightarrow \quad \mu_{v}^{(\mathrm{PTP})}(\mathbf{1}) \tag{5.31}$$

$$\mu_{v}^{(\mathrm{BP})}(*) \quad \leftrightarrow \quad \mu_{v}^{(\mathrm{PTP})}(*). \tag{5.32}$$

Proof: Note that based on Proposition 5, condition $\rho_{c\to v}^{(\mathrm{BP})}(\mathbf{L}*) = \rho_{c\to v}^{(\mathrm{BP})}(\bar{\mathbf{L}}*) = \rho_{c\to v}^{(\mathrm{BP})}(**)$ holds in every BP iteration. From the proof of Proposition 5, it also holds in every BP iteration that

$$\lambda_{v\to c}^{\mathrm{norm}(\mathrm{BP})}(\mathbf{L}*) + \lambda_{v\to c}^{\mathrm{norm}(\mathrm{BP})}(**) = \lambda_{v\to c}^{\mathrm{norm}(\mathrm{BP})}(\mathbf{LL}). \tag{5.33}$$

Now we will prove this theorem by first proving that the "left correspondence" ((5.25) to (5.27)) implies the "right correspondence" ((5.28) and (5.29)) and conversely that the "right correspondence" implies the "left correspondence", whereby proving the correspondence in the passed messages. We then prove the summary correspondence ((5.30) to (5.32)).

First suppose that left correspondence holds, namely that $\lambda_{v\to c}^{\mathrm{norm}(\mathrm{BP})}(\mathbf{L}*) = [L_{v,c} = 0] \cdot \lambda_{v\to c}^{\mathrm{norm}(\mathrm{PTP})}(\mathbf{0}) + [L_{v,c} = 1] \cdot \lambda_{v\to c}^{\mathrm{norm}(\mathrm{PTP})}(\mathbf{1})$, $\lambda_{v\to c}^{\mathrm{norm}(\mathrm{BP})}(\bar{\mathbf{L}}*) = [L_{v,c} = 0] \cdot \lambda_{v\to c}^{\mathrm{norm}(\mathrm{PTP})}(\mathbf{1}) + [L_{v,c} = 1] \cdot \lambda_{v\to c}^{\mathrm{norm}(\mathrm{PTP})}(\mathbf{0})$, and $\lambda_{v\to c}^{\mathrm{norm}(\mathrm{BP})}(**) = \lambda_{v\to c}^{\mathrm{norm}(\mathrm{PTP})}(*)$. Following PTP message-updating equations (4.43) to (4.45), we have

$$\begin{aligned}
& \rho_{c\to v}^{\mathrm{norm}(\mathrm{PTP})}(\mathbf{0}) + \rho_{c\to v}^{\mathrm{norm}(\mathrm{PTP})}(\mathbf{1}) \\
\overset{(a)}{=} \; & \rho_{c\to v}^{(\mathrm{PTP})}(\mathbf{0}) + \rho_{c\to v}^{(\mathrm{PTP})}(\mathbf{1}) \\
= \; & [L_{v,c} = 0] \cdot \prod_{u\in V(c)\setminus\{v\}: L_{u,c}=1} \lambda_{u\to c}^{\mathrm{norm}(\mathrm{PTP})}(\mathbf{0}) \prod_{u\in V(c)\setminus\{v\}: L_{u,c}=0} \lambda_{u\to c}^{\mathrm{norm}(\mathrm{PTP})}(\mathbf{1}) \\
& + [L_{v,c} = 1] \cdot \prod_{u\in V(c)\setminus\{v\}: L_{u,c}=1} \lambda_{u\to c}^{\mathrm{norm}(\mathrm{PTP})}(\mathbf{0}) \prod_{u\in V(c)\setminus\{v\}: L_{u,c}=0} \lambda_{u\to c}^{\mathrm{norm}(\mathrm{PTP})}(\mathbf{1}) \\
= \; & \prod_{u\in V(c)\setminus\{v\}: L_{u,c}=1} \lambda_{u\to c}^{\mathrm{norm}(\mathrm{PTP})}(\mathbf{0}) \prod_{u\in V(c)\setminus\{v\}: L_{u,c}=0} \lambda_{u\to c}^{\mathrm{norm}(\mathrm{PTP})}(\mathbf{1})
\end{aligned}$$

$$
\begin{aligned}
&= \prod_{u\in V(c)\setminus\{v\}:L_{u,c}=1} \Big([L_{u,c}=0]\cdot\lambda_{u\to c}^{\mathrm{norm(PTP)}}(\mathbf{1}) + [L_{u,c}=1]\cdot\lambda_{u\to c}^{\mathrm{norm(PTP)}}(\mathbf{0})\Big) \\
&\quad \times \prod_{u\in V(c)\setminus\{v\}:L_{u,c}=0} \Big([L_{u,c}=0]\cdot\lambda_{u\to c}^{\mathrm{norm(PTP)}}(\mathbf{1}) + [L_{u,c}=1]\cdot\lambda_{u\to c}^{\mathrm{norm(PTP)}}(\mathbf{0})\Big) \\
&= \prod_{u\in V(c)\setminus\{v\}} \Big([L_{u,c}=0]\cdot\lambda_{u\to c}^{\mathrm{norm(PTP)}}(\mathbf{1}) + [L_{u,c}=1]\cdot\lambda_{u\to c}^{\mathrm{norm(PTP)}}(\mathbf{0})\Big) \\
&\overset{(b)}{=} \prod_{u\in V(c)\setminus\{v\}} \lambda_{u\to c}^{\mathrm{norm(BP)}}(\bar{\mathbf{L}}*) \\
&= \rho_{c\to v}^{(\mathrm{BP})}(\mathbf{LL})
\end{aligned}
$$

where equality (a) is due to the fact that $\rho_{c\to v}^{\mathrm{norm(PTP)}} = \rho_{c\to v}^{(\mathrm{PTP})}$ as is shown in the proof of Theorem 2, equality (b) is due to the assumed left correspondence.

Similarly, we have

$$
\begin{aligned}
\rho_{c\to v}^{\mathrm{norm(PTP)}}(*) &= \rho_{c\to v}^{(\mathrm{PTP})}(*) \\
&= 1 - \prod_{u\in V(c)\setminus\{v\}:L_{u,c}=1} \lambda_{u\to c}^{\mathrm{norm(PTP)}}(\mathbf{0}) \prod_{u\in V(c)\setminus\{v\}:L_{u,c}=0} \lambda_{u\to c}^{\mathrm{norm(PTP)}}(\mathbf{1}) \\
&= 1 - \prod_{u\in V(c)\setminus\{v\}} \lambda_{u\to c}^{\mathrm{norm(BP)}}(\bar{\mathbf{L}}*) \\
&\overset{(c)}{=} \prod_{u\in V(c)\setminus\{v\}} (\lambda_{u\to c}^{\mathrm{norm(BP)}}(\mathbf{L}*) + \lambda_{u\to c}^{\mathrm{norm(BP)}}(\bar{\mathbf{L}}*) + \lambda_{u\to c}^{\mathrm{norm(BP)}}(**)) - \prod_{u\in V(c)\setminus\{v\}} \lambda_{u\to c}^{\mathrm{norm(BP)}}(\bar{\mathbf{L}}*) \\
&= \rho_{c\to v}^{(\mathrm{BP})}(\mathbf{L}*),
\end{aligned}
$$

where equality (c) is due to the fact that $\lambda_{u\to c}^{\mathrm{norm(BP)}}(\mathbf{L}*) + \lambda_{u\to c}^{\mathrm{norm(BP)}}(\bar{\mathbf{L}}*) + \lambda_{u\to c}^{\mathrm{norm(BP)}}(**) = 1$.

Thus we proved that if the left correspondence holds, then the right correspondence holds.

Now suppose that the right correspondence holds, namely that $\rho_{c\to v}^{(\mathrm{BP})}(\mathbf{L}*) = \rho_{c\to v}^{\mathrm{norm(PTP)}}(*)$, and $\rho_{c\to v}^{(\mathrm{BP})}(\mathbf{LL}) = \rho_{c\to v}^{\mathrm{norm(PTP)}}(\mathbf{0}) + \rho_{c\to v}^{\mathrm{norm(PTP)}}(\mathbf{1})$. We then have

$$
\begin{aligned}
\rho_{c\to v}^{(\mathrm{BP})}(\mathbf{L}*) + \rho_{c\to v}^{(\mathrm{BP})}(\mathbf{LL}) &= \rho_{c\to v}^{\mathrm{norm(PTP)}}(*) + \rho_{c\to v}^{\mathrm{norm(PTP)}}(\mathbf{0}) + \rho_{c\to v}^{\mathrm{norm(PTP)}}(\mathbf{1}) \\
&= 1.
\end{aligned}
$$

Following PTP message-update equations (4.40) to (4.42), we have

$$
\begin{aligned}
& [L_{v,c}=0]\cdot\lambda_{v\to c}^{(\mathrm{PTP})}(\mathbf{0})+[L_{v,c}=1]\cdot\lambda_{v\to c}^{(\mathrm{PTP})}(\mathbf{1}) \\
= \quad & [L_{v,c}=0]\cdot\left(\prod_{b\in C(v)\backslash\{c\}}(\rho_{b\to v}^{\mathrm{norm(PTP)}}(\mathbf{0})+\rho_{b\to v}^{\mathrm{norm(PTP)}}(*))-\epsilon\prod_{b\in C(v)\backslash\{c\}}\rho_{b\to v}^{\mathrm{norm(PTP)}}(*)\right) \\
& +[L_{v,c}=1]\cdot\left(\prod_{b\in C(v)\backslash\{c\}}(\rho_{b\to v}^{\mathrm{norm(PTP)}}(\mathbf{1})+\rho_{b\to v}^{\mathrm{norm(PTP)}}(*))-\epsilon\prod_{b\in C(v)\backslash\{c\}}\rho_{b\to v}^{\mathrm{norm(PTP)}}(*)\right) \\
= \quad & [L_{v,c}=0]\cdot\prod_{b\in C(v)\backslash\{c\}}(\rho_{b\to v}^{\mathrm{norm(PTP)}}(\mathbf{0})+\rho_{b\to v}^{\mathrm{norm(PTP)}}(*)) \\
& +[L_{v,c}=1]\cdot\prod_{b\in C(v)\backslash\{c\}}(\rho_{b\to v}^{\mathrm{norm(PTP)}}(\mathbf{1})+\rho_{b\to v}^{\mathrm{norm(PTP)}}(*))-\epsilon\prod_{b\in C(v)\backslash\{c\}}\rho_{b\to v}^{\mathrm{norm(PTP)}}(*) \\
\overset{(4.57)}{=} \quad & [L_{v,c}=0]\cdot\prod_{b\in C_c^{\mathrm{s}}(v)}(\rho_{b\to v}^{\mathrm{norm(PTP)}}(\mathbf{0})+\rho_{b\to v}^{\mathrm{norm(PTP)}}(*))\cdot\prod_{b\in C_c^{\mathrm{u}}(v)}\rho_{b\to v}^{\mathrm{norm(PTP)}}(*) \\
& +[L_{v,c}=1]\cdot\prod_{b\in C_c^{\mathrm{s}}(v)}(\rho_{b\to v}^{\mathrm{norm(PTP)}}(\mathbf{1})+\rho_{b\to v}^{\mathrm{norm(PTP)}}(*))\cdot\prod_{b\in C_c^{\mathrm{u}}(v)}\rho_{b\to v}^{\mathrm{norm(PTP)}}(*) \\
& -\epsilon\prod_{b\in C(v)\backslash\{c\}}\rho_{b\to v}^{\mathrm{norm(PTP)}}(*) \\
\overset{(4.56)}{=} \quad & [L_{v,c}=0]\cdot\prod_{b\in C_c^{\mathrm{u}}(v)}\rho_{b\to v}^{\mathrm{norm(PTP)}}(*)+[L_{v,c}=1]\cdot\prod_{b\in C_c^{\mathrm{u}}(v)}\rho_{b\to v}^{\mathrm{norm(PTP)}}(*)-\epsilon\prod_{b\in C(v)\backslash\{c\}}\rho_{b\to v}^{\mathrm{norm(PTP)}}(*) \\
= \quad & \prod_{b\in C_c^{\mathrm{u}}(v)}\rho_{b\to v}^{\mathrm{norm(PTP)}}(*)-\epsilon\prod_{b\in C(v)\backslash\{c\}}\rho_{b\to v}^{\mathrm{norm(PTP)}}(*) \\
= \quad & \prod_{b\in C_c^{\mathrm{u}}(v)}\rho_{b\to v}^{\mathrm{norm(PTP)}}(*)\left(1-\epsilon\prod_{b\in C_c^{\mathrm{s}}(v)}\rho_{b\to v}^{\mathrm{norm(PTP)}}(*)\right) \\
\overset{(d)}{=} \quad & \prod_{b\in C_c^{\mathrm{u}}(v)}\rho_{b\to v}^{(\mathrm{BP})}(\mathbf{L}*)\left(1-\epsilon\prod_{b\in C_c^{\mathrm{s}}(v)}\rho_{b\to v}^{(\mathrm{BP})}(\mathbf{L}*)\right) \\
\overset{(e)}{=} \quad & \prod_{b\in C_c^{\mathrm{u}}(v)}\rho_{b\to v}^{(\mathrm{BP})}(\mathbf{L}*)\left(\prod_{b\in C_c^{\mathrm{s}}(v)}(\rho_{b\to v}^{(\mathrm{BP})}(\mathbf{L}*)+\rho_{b\to v}^{(\mathrm{BP})}(\mathbf{L}\mathbf{L}))-\epsilon\prod_{b\in C_c^{\mathrm{s}}(v)}\rho_{b\to v}^{(\mathrm{BP})}(\mathbf{L}*)\right) \\
\overset{(f)}{=} \quad & \prod_{b\in C_c^{\mathrm{u}}(v)}\rho_{b\to v}^{(\mathrm{BP})}(\bar{\mathbf{L}}*)\left(\prod_{b\in C_c^{\mathrm{s}}(v)}(\rho_{b\to v}^{(\mathrm{BP})}(\mathbf{L}*)+\rho_{b\to v}^{(\mathrm{BP})}(\mathbf{L}\mathbf{L}))-\epsilon\prod_{b\in C_c^{\mathrm{s}}(v)}\rho_{b\to v}^{(\mathrm{BP})}(\mathbf{L}*)\right) \\
= \quad & \lambda_{v\to c}^{(\mathrm{BP})}(\mathbf{L}*)
\end{aligned}
$$

where equality (d) is due to the assumed right correspondence, equality (e) is due to the fact that

$\rho_{c\to v}^{(\mathrm{BP})}(\mathbf{L}*)+\rho_{c\to v}^{(\mathrm{BP})}(\mathbf{LL})=1$, and equality (f) is due to that the condition $\rho_{b\to v}^{(\mathrm{BP})}(\mathbf{L}*)=\rho_{b\to v}^{(\mathrm{BP})}(\bar{\mathbf{L}}*)$ is satisfied in every iteration. We will denote this result by (A).

Similarly, we have

$$\begin{aligned}
& [L_{v,c}=0]\cdot\lambda_{v\to c}^{(\mathrm{PTP})}(\mathbf{1})+[L_{v,c}=1]\cdot\lambda_{v\to c}^{(\mathrm{PTP})}(\mathbf{0}) \\
= \; & [L_{v,c}=0]\cdot\left(\prod_{b\in C(v)\setminus\{c\}}(\rho_{b\to v}^{\mathrm{norm(PTP)}}(\mathbf{1})+\rho_{b\to v}^{\mathrm{norm(PTP)}}(*))-\epsilon\prod_{b\in C(v)\setminus\{c\}}\rho_{b\to v}^{\mathrm{norm(PTP)}}(*)\right) \\
& +[L_{v,c}=1]\cdot\left(\prod_{b\in C(v)\setminus\{c\}}(\rho_{b\to v}^{\mathrm{norm(PTP)}}(\mathbf{0})+\rho_{b\to v}^{\mathrm{norm(PTP)}}(*))-\epsilon\prod_{b\in C(v)\setminus\{c\}}\rho_{b\to v}^{\mathrm{norm(PTP)}}(*)\right) \\
= \; & \lambda_{v\to c}^{(\mathrm{BP})}(\bar{\mathbf{L}}*).
\end{aligned}$$

We will denote this result by (B).

Finally, we have

$$\begin{aligned}
\lambda_{v\to c}^{(\mathrm{PTP})}(*) &= \epsilon\prod_{b\in C(v)\setminus\{c\}}\rho_{b\to v}^{\mathrm{norm(PTP)}}(*) \\
&= \epsilon\prod_{b\in C(v)\setminus\{c\}}\rho_{b\to v}^{(\mathrm{BP})}(**) \\
&= \lambda_{v\to c}^{(\mathrm{BP})}(**).
\end{aligned}$$

We will denote this result by (C).

Combining results of (A), (B) and (C), we have

$$\lambda_{v\to c}^{(\mathrm{PTP})}(\mathbf{0})+\lambda_{v\to c}^{(\mathrm{PTP})}(\mathbf{1})+\lambda_{v\to c}^{(\mathrm{PTP})}(*)=\lambda_{v\to c}^{(\mathrm{BP})}(\mathbf{L}*)+\lambda_{v\to c}^{(\mathrm{BP})}(\bar{\mathbf{L}}*)+\lambda_{v\to c}^{(\mathrm{BP})}(**).$$

That is, the scaling constant for normalizing $(\lambda_{v\to c}^{(\mathrm{PTP})}(\mathbf{0}),\lambda_{v\to c}^{(\mathrm{PTP})}(\mathbf{1}),\lambda_{v\to c}^{(\mathrm{PTP})}(*))$ and that for normalizing $(\lambda_{v\to c}^{(\mathrm{BP})}(\mathbf{L}*),\lambda_{v\to c}^{(\mathrm{BP})}(\bar{\mathbf{L}}*),\lambda_{v\to c}^{(\mathrm{BP})}(**))$ are identical. Therefore, result (A), (B) and (C)

respectively translate to

$$\begin{aligned}
[L_{v,c}=0]\cdot\lambda_{v\to c}^{\mathrm{norm(PTP)}}(\mathbf{0})+[L_{v,c}=1]\cdot\lambda_{v\to c}^{\mathrm{norm(PTP)}}(\mathbf{1}) &= \lambda_{v\to c}^{\mathrm{norm(BP)}}(\mathbf{L}*)\\
[L_{v,c}=0]\cdot\lambda_{v\to c}^{\mathrm{norm(PTP)}}(\mathbf{1})+[L_{v,c}=1]\cdot\lambda_{v\to c}^{\mathrm{norm(PTP)}}(\mathbf{0}) &= \lambda_{v\to c}^{\mathrm{norm(BP)}}(\bar{\mathbf{L}}*)\\
\lambda_{v\to c}^{\mathrm{norm(PTP)}}(*) &= \lambda_{v\to c}^{\mathrm{norm(BP)}}(**).
\end{aligned}$$

At this point we have proved the correspondence between the passed messages in BP and those in weighted PTP.

We now prove the summary correspondence. Following the PTP message-update equations (4.46) to (4.48), we have

$$\begin{aligned}
\mu_v^{(\mathrm{PTP})}(\mathbf{0}) &= \prod_{c\in C(v)}\left(\rho_{c\to v}^{\mathrm{norm(PTP)}}(\mathbf{0})+\rho_{c\to v}^{\mathrm{norm(PTP)}}(*)\right)-\epsilon\prod_{c\in C(v)}\rho_{c\to v}^{\mathrm{norm(PTP)}}(*)\\
&= \prod_{c\in C^1(v)}\left(\rho_{c\to v}^{\mathrm{norm(PTP)}}(\mathbf{0})+\rho_{c\to v}^{\mathrm{norm(PTP)}}(*)\right)\prod_{c\in C^0(v)}\left(\rho_{c\to v}^{\mathrm{norm(PTP)}}(\mathbf{0})+\rho_{c\to v}^{\mathrm{norm(PTP)}}(*)\right)\\
&\quad -\epsilon\prod_{c\in C(v)}\rho_{c\to v}^{\mathrm{norm(PTP)}}(*)\\
&\overset{(4.56),(4.57)}{=} \prod_{c\in C^1(v)}\rho_{c\to v}^{\mathrm{norm(PTP)}}(*)-\epsilon\prod_{c\in C(v)}\rho_{c\to v}^{\mathrm{norm(PTP)}}(*)\\
&= \prod_{c\in C^1(v)}\rho_{c\to v}^{\mathrm{norm(PTP)}}(*)\left(1-\epsilon\prod_{c\in C^0(v)}\rho_{c\to v}^{\mathrm{norm(PTP)}}(*)\right)\\
&= \prod_{c\in C^1(v)}\rho_{c\to v}^{(\mathrm{BP})}(\bar{\mathbf{L}}*)\left(\prod_{c\in C^0(v)}(\rho_{c\to v}^{(\mathrm{BP})}(\mathbf{LL})+\rho_{c\to v}^{(\mathrm{BP})}(\mathbf{L}*)-\epsilon\prod_{c\in C^0(v)}\rho_{c\to v}^{(\mathrm{BP})}(\mathbf{L}*)\right)\\
&= \mu_v^{(\mathrm{BP})}(\mathbf{0}).
\end{aligned}$$

Following a similar procedure, we have

$$\begin{aligned}
\mu_v^{(\mathrm{PTP})}(\mathbf{1}) &= \prod_{c\in C(v)}(\rho_{c\to v}^{\mathrm{norm(PTP)}}(\mathbf{1})+\rho_{c\to v}^{\mathrm{norm(PTP)}}(*))-\epsilon\prod_{c\in C(v)}\rho_{c\to v}^{\mathrm{norm(PTP)}}(*)\\
&= \prod_{c\in C^0(v)}\rho_{c\to v}^{(\mathrm{BP})}(\bar{\mathbf{L}}*)\left(\prod_{c\in C^1(v)}(\rho_{c\to v}^{(\mathrm{BP})}(\mathbf{LL})+\rho_{c\to v}^{(\mathrm{BP})}(\mathbf{L}*)-\epsilon\prod_{c\in C^1(v)}\rho_{c\to v}^{(\mathrm{BP})}(\mathbf{L}*)\right)\\
&= \mu_v^{(\mathrm{BP})}(\mathbf{1}).
\end{aligned}$$

Finally, we have

$$\mu_v^{(\mathrm{PTP})}(*) \quad = \quad \epsilon \prod_{c \in C(v)} \rho_{c \to v}^{\mathrm{norm}(\mathrm{PTP})}(*) = \epsilon \prod_{c \in C(v)} \rho_{c \to v}^{(\mathrm{BP})}(**) = \mu_v^{(\mathrm{BP})}(*),$$

which proves the summary correspondence. ■

5.3 State-Decoupled BP

In this section, we will consider reducing PTP from BP for 3-COL problems, where we only focus on the non-weighted version of PTP, namely that each weighting function ω_v is defined as

$$\omega_v(a|b) := [a = b]. \tag{5.34}$$

This gives the form of BP messages in the form specified in the following lemma, easily obtainable from BP update equations (5.4) to (5.6).

Lemma 17 *The BP message-update rule for 3-COL problems is as follow:*

$$\begin{aligned}
\lambda_{v \to c}(\mathbf{i}, \mathbf{ij}) \quad &:= \quad \prod_{b \in C(v) \setminus \{c\}} (\rho_{b \to v}(\mathbf{i}, \mathbf{ij}) + \rho_{b \to v}(\mathbf{i}, \mathbf{ik}) + \rho_{b \to v}(\mathbf{i}, \mathbf{ijk})) \\
&\quad - \prod_{b \in C(v) \setminus \{c\}} (\rho_{b \to v}(\mathbf{i}, \mathbf{ij}) + \rho_{b \to v}(\mathbf{i}, \mathbf{ijk})) && (5.35) \\
\lambda_{v \to c}(\mathbf{i}, \mathbf{ijk}) \quad &:= \quad \prod_{b \in C(v) \setminus \{c\}} (\rho_{b \to v}(\mathbf{i}, \mathbf{ij}) + \rho_{b \to v}(\mathbf{i}, \mathbf{ik}) + \rho_{b \to v}(\mathbf{i}, \mathbf{ijk})) \\
&\quad - \prod_{b \in C(v) \setminus \{c\}} (\rho_{b \to v}(\mathbf{i}, \mathbf{ij}) + \rho_{b \to v}(\mathbf{i}, \mathbf{ijk})) \\
&\quad - \prod_{b \in C(v) \setminus \{c\}} (\rho_{b \to v}(\mathbf{i}, \mathbf{ik}) + \rho_{b \to v}(\mathbf{i}, \mathbf{ijk})) + \prod_{b \in C(v) \setminus \{c\}} \rho_{b \to v}(\mathbf{i}, \mathbf{ijk}) && (5.36) \\
\lambda_{v \to c}(\mathbf{ij}, \mathbf{ij}) \quad &:= \quad \prod_{b \in C(v) \setminus \{c\}} (\rho_{b \to v}(\mathbf{ij}, \mathbf{ij}) + \rho_{b \to v}(\mathbf{ij}, \mathbf{ijk})) && (5.37) \\
\lambda_{v \to c}(\mathbf{ij}, \mathbf{ijk}) \quad &:= \quad \prod_{b \in C(v) \setminus \{c\}} (\rho_{b \to v}(\mathbf{ij}, \mathbf{ij}) + \rho_{b \to v}(\mathbf{ij}, \mathbf{ijk})) - \prod_{b \in C(v) \setminus \{c\}} \rho_{b \to v}(\mathbf{ij}, \mathbf{ijk}) && (5.38) \\
\lambda_{v \to c}(\mathbf{ijk}, \mathbf{ijk}) \quad &:= \quad \prod_{b \in C(v) \setminus \{c\}} \rho_{b \to v}(\mathbf{ijk}, \mathbf{ijk}) && (5.39)
\end{aligned}$$

$$\rho_{c\to v}(\mathbf{i},\mathbf{ij}) := \lambda_{V(c)\setminus\{v\}\to c}(\mathbf{k},\mathbf{jk}) \tag{5.40}$$

$$\rho_{c\to v}(\mathbf{i},\mathbf{ijk}) := \lambda_{V(c)\setminus\{v\}\to c}(\mathbf{jk},\mathbf{jk}) \tag{5.41}$$

$$\rho_{c\to v}(\mathbf{ij},\mathbf{ij}) := \lambda_{V(c)\setminus\{v\}\to c}(\mathbf{k},\mathbf{ijk}) \tag{5.42}$$

$$\begin{aligned}\rho_{c\to v}(\mathbf{ij},\mathbf{ijk}) := {} & \lambda_{V(c)\setminus\{v\}\to c}(\mathbf{ij},\mathbf{ijk}) + \lambda_{V(c)\setminus\{v\}\to c}(\mathbf{jk},\mathbf{ijk}) + \lambda_{V(c)\setminus\{v\}\to c}(\mathbf{ik},\mathbf{ijk}) \\ & +\lambda_{V(c)\setminus\{v\}\to c}(\mathbf{ijk},\mathbf{ijk})\end{aligned} \tag{5.43}$$

$$\begin{aligned}\rho_{c\to v}(\mathbf{ijk},\mathbf{ijk}) := {} & \lambda_{V(c)\setminus\{v\}\to c}(\mathbf{ij},\mathbf{ijk}) + \lambda_{V(c)\setminus\{v\}\to c}(\mathbf{jk},\mathbf{ijk}) + \lambda_{V(c)\setminus\{v\}\to c}(\mathbf{ik},\mathbf{ijk}) \\ & +\lambda_{V(c)\setminus\{v\}\to c}(\mathbf{ijk},\mathbf{ijk})\end{aligned} \tag{5.44}$$

$$\begin{aligned}\mu_v(\mathbf{i}) := {} & \prod_{c\in C(v)} (\rho_{c\to v}(\mathbf{i},\mathbf{ij}) + \rho_{c\to v}(\mathbf{i},\mathbf{ik}) + \rho_{c\to v}(\mathbf{i},\mathbf{ijk})) \\ & - \prod_{c\in C(v)} (\rho_{c\to v}(\mathbf{i},\mathbf{ij}) + \rho_{c\to v}(\mathbf{i},\mathbf{ijk})) \\ & - \prod_{c\in C(v)} (\rho_{c\to v}(\mathbf{i},\mathbf{ik}) + \rho_{c\to v}(\mathbf{i},\mathbf{ijk})) + \prod_{c\in C(v)} \rho_{c\to v}(\mathbf{i},\mathbf{ijk})\end{aligned} \tag{5.45}$$

$$\mu_v(\mathbf{ij}) := \prod_{c\in C(v)} (\rho_{c\to v}(\mathbf{ij},\mathbf{ij}) + \rho_{c\to v}(\mathbf{ij},\mathbf{ijk})) - \prod_{c\in C(v)} \rho_{c\to v}(\mathbf{ij},\mathbf{ijk}) \tag{5.46}$$

$$\mu_v(\mathbf{ijk}) := \prod_{c\in C(v)} \rho_{c\to v}(\mathbf{ijk},\mathbf{ijk}). \tag{5.47}$$

Before we begin to consider the BP-to-PTP reduction for 3-COL problems, it is helpful to take a closer look at the BP-to-PTP reduction mechanism for k-SAT problems.

In Theorem 3, one may notice the two conditions governing the BP-to-PTP reduction for k-SAT problems, namely, the initialization condition and the normalization condition. It is arguable that the normalization condition imposed on the BP messages, although serving to simplify the form of BP messages and possibly to alter the interpretation of the messages, does not have a critical impact on the message-passing dynamics. This is because the normalization condition merely involves a scaling operation, without which BP messages and PTP messages for k-SAT would still be equivalent up to a scaling factor. On the other hand, the initialization condition in Theorem 3 plays an important role on the message-passing dynamics. In essence, the initialization condition assures that any right message depends only on the right state it

involves. Using the "intention-command" analogy, in which one views each right state as storing the "command" sent from a constraint and each left state as storing the "intention" of a variable, this condition simply restricts that the *distribution* of the command sent to any variable does *not* depend on the intention of the variable. It is remarkable that this interpretation of the initialization condition in Theorem 3 (or (5.20)) is consistent with the PTP message-passing rule, in which any right message (i.e., outgoing distribution of command) sent to a variable is independent of (or, not a function of,) the incoming intention from that variable. This is however not the case for the right messages of BP in general.

We are then motivated to formalize this condition for general CSPs as what we call the "state-decoupling" condition and impose it on the right messages of BP, so as to achieve a consistency with PTP. It is intuitively sensible that such a consistency is needed in the reduction of PTP from BP.

Definition 2 (State-Decoupling Condition) *For an arbitrary CSP and at any given iteration, the BP messages based on the MRF formalism defined by (5.1), (5.2), and (5.3) are said to satisfy the state-decoupling condition if for every $(v-c)$, the right message $\rho_{c\to v}(s_{v,c})$ is only a function of the right state $s^R_{v,c}$, namely, if for any fixed $s^R_{v,c} \in (\chi^*)^{\{v\}}$ and any $s^L_{v,c} \subset s^R_{v,c}$, $\rho_{c\to v}(s^L_{v,c}, s^R_{v,c}) = \rho_{c\to v}(s^R_{v,c}, s^R_{v,c})$.*

It is clear that the initialization condition for BP-to-PTP reduction for k-SAT in Theorem 3 is equivalent to this condition, where we note that the condition in Theorem 3 only puts restrictions on the right messages with right state equal to $*$, since for the remaining case with right state equal to **L** this condition is trivially satisfied.

It is interesting to observe, as shown in Proposition 5, that for k-SAT problems, as long as the state-decoupling condition is imposed in the initialization of the BP messages, the condition is preserved in every iteration. This serves as the basis for BP to reduce to PTP as shown in Theorem 3 and its proof. For 3-COL problems, however, the corresponding result to Proposition 5 does not hold.

Lemma 18 *For 3-COL problems, if the state-decoupling condition holds for BP messages both in iteration l and in iteration $l+1$, then the right message in iteration l must satisfy for every $(v-c)$*

$$\rho_{c\to v}(s^L, s^R) = 0$$

as long as right state $s^R \neq \mathbf{123}$.

Proof: In 3-COL problems, the state-decoupling condition can be expressed as

$$\begin{aligned}
\rho_{c\to v}(\mathbf{i},\mathbf{ij}) &= \rho_{c\to v}(\mathbf{ij},\mathbf{ij}) \\
\rho_{c\to v}(\mathbf{i},\mathbf{ijk}) &= \rho_{c\to v}(\mathbf{ij},\mathbf{ijk}) = \rho_{c\to v}(\mathbf{ijk},\mathbf{ijk}).
\end{aligned}$$

Note that we only need to prove the lemma for s^R being a pair of assignments, since when s^R is a singleton, all right messages equal 0 by the construction of the MRF and Lemma 17 describing the BP message-update rule for 3-COL.

In iteration $l+1$, following 3-COL message-update equations (5.35) to (5.44) and using a superscript to denote the iteration number, we have

$$\begin{aligned}
\rho^{(l+1)}_{c\to v}(\mathbf{i},\mathbf{ij}) &= \lambda^{(l+1)}_{V(c)\backslash\{v\}\to c}(\mathbf{k},\mathbf{jk}) \\
&= \prod_{b\in C(V(c)\backslash\{v\})\backslash\{c\}} \left(\rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{jk}) + \rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{ik}) + \rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{ijk})\right) \\
&\quad - \prod_{b\in C(V(c)\backslash\{v\})\backslash\{c\}} \left(\rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{jk}) + \rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{ijk})\right), \qquad (5.48) \\
\rho^{(l+1)}_{c\to v}(\mathbf{ij},\mathbf{ij}) &= \lambda^{(l+1)}_{V(c)\backslash\{v\}\to c}(\mathbf{k},\mathbf{ijk}) \\
&= \prod_{b\in C(V(c)\backslash\{v\})\backslash\{c\}} \left(\rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{jk}) + \rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{ik}) + \rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{ijk})\right) \\
&\quad - \prod_{b\in C(V(c)\backslash\{v\})\backslash\{c\}} \left(\rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{ik}) + \rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{ijk})\right) \\
&\quad - \prod_{b\in C(V(c)\backslash\{v\})\backslash\{c\}} \left(\rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{jk}) + \rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{ijk})\right) \\
&\quad + \prod_{b\in C(V(c)\backslash\{v\})\backslash\{c\}} \rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{ijk}). \qquad (5.49)
\end{aligned}$$

Now suppose that the state-decoupling condition as expressed above can be satisfied both in iteration l and in iteration $l+1$. Then we may equate the right-hand sides of (5.48) and (5.49), namely,

$$\begin{aligned}
&\prod_{b\in C(V(c)\backslash\{v\})\backslash\{c\}} \left(\rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{jk}) + \rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{ik}) + \rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{ijk})\right) \\
&- \prod_{b\in C(V(c)\backslash\{v\})\backslash\{c\}} \left(\rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{jk}) + \rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{ijk})\right) \\
= &\prod_{b\in C(V(c)\backslash\{v\})\backslash\{c\}} \left(\rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{jk}) + \rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{ik}) + \rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{ijk})\right) \\
&- \prod_{b\in C(V(c)\backslash\{v\})\backslash\{c\}} \left(\rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{ik}) + \rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{ijk})\right) \\
&- \prod_{b\in C(V(c)\backslash\{v\})\backslash\{c\}} \left(\rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{jk}) + \rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{ijk})\right) + \prod_{b\in C(V(c)\backslash\{v\})\backslash\{c\}} \rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{ijk}),
\end{aligned}$$

which implies

$$\begin{aligned}
&\prod_{b\in C(V(c)\backslash\{v\})\backslash\{c\}} \left(\rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{jk}) + \rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{ijk})\right) \\
= &\prod_{b\in C(V(c)\backslash\{v\})\backslash\{c\}} \left(\rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{ik}) + \rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{ijk})\right) \\
&+ \prod_{b\in C(V(c)\backslash\{v\})\backslash\{c\}} \left(\rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{jk}) + \rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{ijk})\right) - \prod_{b\in C(V(c)\backslash\{v\})\backslash\{c\}} \rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{ijk}).
\end{aligned}$$

Since every right message must be non-negative, when the state-decoupling condition is satisfied in iteration l, the only way to make the above equality hold is the case where

$$\rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{k},\mathbf{ik}) = 0.$$

Under the state-decoupling condition, this also means $\rho^{(l)}_{b\to V(c)\backslash\{v\}}(\mathbf{ik},\mathbf{ik}) = 0$. Thus we establish this lemma. ■

This lemma suggests that when the BP messages satisfy the state-decoupling condition in two consecutive iterations, then the right messages must take a trivial form — equal to

$[s^R = \mathbf{123}]$ up to scale, and contain no information.

At this point, one is left with either the option of concluding that PTP (or SP) is *not* an instance of BP for 3-COL problems (and hence for general CSPs) or the option of doubting the usefulness of the state-decoupling condition in BP-to-SP reduction. In the remainder of this section, we will clear this doubt and assert the usefulness of the state-decoupling condition by showing that when the state-decoupling condition is *manually* imposed on the BP messages in each iteration, BP still reduces to PTP for 3-COL problems. That will allow us to conclude that PTP (or SP) is not a special case of BP.

To force the state-decoupling condition to be satisfied in each BP iteration, now we modify the message-passing rule of BP on the Forney graph representation of general CSPs, and introduce a "new" message-passing procedure which we refer to as the *state-decoupled BP* or SDBP. We note that introducing this "new" message-passing procedure is solely for the purpose of verifying the usefulness of the state-decoupling condition and hopefully arriving at a unified reduction mechanism for PTP to reduce from BP (or more precisely from SDBP). Beyond this purpose, we have no intention to justify the introduction of SDBP.

Identical to BP at local function vertices, SDBP differs from BP in that messages passed from the right functions need an additional processing (so that the state-decoupling condition is satisfied) before they are passed to the left functions. In SDBP, there are three kinds of messages: *right message* $\rho_{c\to v}$ is computed at right function f_c to pass along the edge to g_v; *state-decoupled right message* $\rho^*_{c\to v}$ is computed at the edge connecting f_c and g_v, which satisfies the state-decoupling condition, computed only based on the right message $\rho_{c\to v}$ on the same edge and to be passed to left function g_v; *left message* $\lambda_{v\to c}$ is computed at the left function g_v to pass along the edge connecting to f_c. The precise definition of SDBP message-update rule is given next.

Definition 3 *The SDBP message-update rule is defined as follows.*

$$\lambda_{v\to c}(s^L_{v,c}, s^R_{v,c}) := \sum_{s^R_{v,C(v)\setminus\{c\}}} \omega_v \left(s^L_{v,c} \middle| \bigcap_{b\in C(v)} s^R_{v,b} \right) \cdot \prod_{b\in C(v)\setminus\{c\}} \rho^*_{b\to v}(s^R_{v,b}) \tag{5.50}$$

$$\rho^*_{c\to v}(s^R_{v,c}) := \delta \cdot \rho_{c\to v}(s^R_{v,c}, s^R_{v,c}) \tag{5.51}$$

$$\rho_{c\to v}(s^L_{v,c}, s^R_{v,c}) := \sum_{s^L_{V(c)\setminus\{v\},c}} \left[s^R_{v,c} = \mathtt{F}_c(s^L_{V(c)\setminus\{v\},c}) \right] \prod_{u\in V(c)\setminus\{v\}} \lambda_{u\to c}(s^L_{u,c}, \mathtt{F}_c(s^L_{V(c)\setminus\{u\},c})) \tag{5.52}$$

$$\mu_v(y_v) := \sum_{s^R_{v,C(v)}} \omega_v \left(y_v \middle| \bigcap_{c\in C(v)} s^R_{v,c} \right) \prod_{c\in C(v)} \rho^*_{c\to v}(s^R_{v,c}) \tag{5.53}$$

where $\delta = 1/\sum_{s^R_{v,c}\in(\chi^*)^{\{v\}}} \rho_{c\to v}(s^R_{v,c}, s^R_{v,c})$.

Comparing this definition with the BP message-update rule in Lemma 13, the following remarks are in order. First, the expression of right messages ρ in terms of left messages λ is identical to that in BP. Second, each state-decoupled message $\rho^*_{c\to v}$ may be regarded as a function of $(s^L_{v,c}, s^R_{v,c})$ but the value of the function only depends the $s^R_{v,c}$ component, namely that the (state-decoupled) right message satisfies the state-decoupling condition. Furthermore, the expression of λ in terms of ρ^* is precisely the same as the expression of λ in terms of ρ in BP[1].

Following this definition, the next lemma summarizes the SDBP message-update rule for 3-COL problems.

Lemma 19 *Let* $\{\omega_v : v \in V\}$ *in 3-COL problems be defined as in (5.34). The SDBP message-update rule is then :*

[1] Although it is possible to formulate SDBP in more compact form by, for example, suppressing ρ and expressing the message-update rule only using ρ^* and λ, we feel the current way of formulating SDBP makes it easier to compare SDBP with BP.

$$\begin{aligned}
\lambda_{v\to c}(\mathbf{i},\mathbf{ij}) &:= \prod_{b\in C(v)\setminus\{c\}} (\rho^*_{b\to v}(\mathbf{ij}) + \rho^*_{b\to v}(\mathbf{ik}) + \rho^*_{b\to v}(\mathbf{ijk})) \\
&\quad - \prod_{b\in C(v)\setminus\{c\}} (\rho^*_{b\to v}(\mathbf{ij}) + \rho^*_{b\to v}(\mathbf{ijk})) && (5.54)\\
\lambda_{v\to c}(\mathbf{i},\mathbf{ijk}) &:= \prod_{b\in C(v)\setminus\{c\}} (\rho^*_{b\to v}(\mathbf{ij}) + \rho^*_{b\to v}(\mathbf{ik}) + \rho^*_{b\to v}(\mathbf{ijk})) - \prod_{b\in C(v)\setminus\{c\}} (\rho^*_{b\to v}(\mathbf{ij}) + \rho^*_{b\to v}(\mathbf{ijk})) \\
&\quad - \prod_{b\in C(v)\setminus\{c\}} (\rho^*_{b\to v}(\mathbf{ik}) + \rho^*_{b\to v}(\mathbf{ijk})) + \prod_{b\in C(v)\setminus\{c\}} \rho^*_{b\to v}(\mathbf{ijk}) && (5.55)\\
\lambda_{v\to c}(\mathbf{ij},\mathbf{ij}) &:= \prod_{b\in C(v)\setminus\{c\}} (\rho^*_{b\to v}(\mathbf{ij}) + \rho^*_{b\to v}(\mathbf{ijk})) && (5.56)\\
\lambda_{v\to c}(\mathbf{ij},\mathbf{ijk}) &:= \prod_{b\in C(v)\setminus\{c\}} (\rho^*_{b\to v}(\mathbf{ij}) + \rho^*_{b\to v}(\mathbf{ijk})) - \prod_{b\in C(v)\setminus\{c\}} \rho^*_{b\to v}(\mathbf{ijk}) && (5.57)\\
\lambda_{v\to c}(\mathbf{ijk},\mathbf{ijk}) &:= \prod_{b\in C(v)\setminus\{c\}} \rho^*_{b\to v}(\mathbf{ijk}) && (5.58)\\
\rho^*_{c\to v}(\mathbf{ij}) &:= \delta\cdot\lambda_{V(c)\setminus\{v\}\to c}(\mathbf{k},\mathbf{ijk}) && (5.59)\\
\rho^*_{c\to v}(\mathbf{ijk}) &:= \delta\cdot\big(\lambda_{V(c)\setminus\{v\}\to c}(\mathbf{ij},\mathbf{ijk}) + \lambda_{V(c)\setminus\{v\}\to c}(\mathbf{ik},\mathbf{ijk}) + \lambda_{V(c)\setminus\{v\}\to c}(\mathbf{jk},\mathbf{ijk}) \\
&\quad + \lambda_{V(c)\setminus\{v\}\to c}(\mathbf{ijk},\mathbf{ijk})\big) && (5.60)\\
\mu_v(\mathbf{i}) &:= \prod_{c\in C(v)} (\rho^*_{c\to v}(\mathbf{ij}) + \rho^*_{c\to v}(\mathbf{ik}) + \rho^*_{c\to v}(\mathbf{ijk})) - \prod_{c\in C(v)} (\rho^*_{c\to v}(\mathbf{ij}) + \rho^*_{c\to v}(\mathbf{ijk})) \\
&\quad - \prod_{c\in C(v)} (\rho^*_{c\to v}(\mathbf{ik}) + \rho^*_{c\to v}(\mathbf{ijk})) + \prod_{c\in C(v)} \rho^*_{c\to v}(\mathbf{ijk}) && (5.61)\\
\mu_v(\mathbf{ij}) &:= \prod_{c\in C(v)} (\rho^*_{c\to v}(\mathbf{ij}) + \rho^*_{c\to v}(\mathbf{ijk})) - \prod_{c\in C(v)} \rho^*_{c\to v}(\mathbf{ijk}) && (5.62)\\
\mu_v(\mathbf{ijk}) &:= \prod_{c\in C(v)} \rho^*_{c\to v}(\mathbf{ijk}), && (5.63)
\end{aligned}$$

where δ is such that

$$\rho^*_{c\to v}(\mathbf{ijk}) + \sum_{\mathbf{ij}} \rho^*_{c\to v}(\mathbf{ij}) = 1.$$

It is now possible to establish a correspondence between PTP and SDBP messages for 3-COL problems.

Theorem 4 *For 3-COL problems, the correspondence between PTP and SDBP message-update*

rules is

$$\lambda_{v\to c}^{(\mathrm{PTP})}(\mathbf{i}) \quad \leftrightarrow \quad \lambda_{v\to c}^{(\mathrm{SDBP})}(\mathbf{i},\mathbf{ijk}) \tag{5.64}$$

$$\lambda_{v\to c}^{(\mathrm{PTP})}(\mathbf{ij}) \quad \leftrightarrow \quad \lambda_{v\to c}^{(\mathrm{SDBP})}(\mathbf{ij},\mathbf{ijk}) \tag{5.65}$$

$$\lambda_{v\to c}^{(\mathrm{PTP})}(\mathbf{ijk}) \quad \leftrightarrow \quad \lambda_{v\to c}^{(\mathrm{SDBP})}(\mathbf{ijk},\mathbf{ijk}) \tag{5.66}$$

$$\rho_{c\to v}^{\mathrm{norm}(\mathrm{PTP})}(\mathbf{ij}) \quad \leftrightarrow \quad \rho_{c\to v}^{*(\mathrm{SDBP})}(\mathbf{ij}) \tag{5.67}$$

$$\rho_{c\to v}^{\mathrm{norm}(\mathrm{PTP})}(\mathbf{ijk}) \quad \leftrightarrow \quad \rho_{c\to v}^{*(\mathrm{SDBP})}(\mathbf{ijk}) \tag{5.68}$$

$$\mu_{v}^{(\mathrm{PTP})}(\mathbf{i}) \quad \leftrightarrow \quad \mu_{v}^{(\mathrm{SDBP})}(\mathbf{i}) \tag{5.69}$$

$$\mu_{v}^{(\mathrm{PTP})}(\mathbf{ij}) \quad \leftrightarrow \quad \mu_{v}^{(\mathrm{SDBP})}(\mathbf{ij}) \tag{5.70}$$

$$\mu_{v}^{(\mathrm{PTP})}(\mathbf{ijk}) \quad \leftrightarrow \quad \mu_{v}^{(\mathrm{SDBP})}(\mathbf{ijk}). \tag{5.71}$$

Proof: We will first prove that if the "right correspondence" (namely that (5.67) and (5.68)) holds, then the "left correspondence" (namely that (5.64) to (5.66)) holds.

Suppose that the right correspondence holds (where the symbol $\leftrightarrow$ in (5.67) and (5.68) is understood as equality). Then

$$\begin{aligned}
\lambda_{v\to c}^{(\mathrm{SDBP})}(\mathbf{i},\mathbf{ijk}) &= \prod_{b\in C(v)\setminus\{c\}} \left(\rho_{b\to v}^{*(\mathrm{SDBP})}(\mathbf{ij}) + \rho_{b\to v}^{*(\mathrm{SDBP})}(\mathbf{ik}) + \rho_{b\to v}^{*(\mathrm{SDBP})}(\mathbf{ijk})\right) \\
&\quad - \prod_{b\in C(v)\setminus\{c\}} \left(\rho_{b\to v}^{*(\mathrm{SDBP})}(\mathbf{ij}) + \rho_{b\to v}^{*(\mathrm{SDBP})}(\mathbf{ijk})\right) \\
&\quad - \prod_{b\in C(v)\setminus\{c\}} \left(\rho_{b\to v}^{*(\mathrm{SDBP})}(\mathbf{ik}) + \rho_{b\to v}^{*(\mathrm{SDBP})}(\mathbf{ijk})\right) + \prod_{b\in C(v)\setminus\{c\}} \rho_{b\to v}^{*(\mathrm{SDBP})}(\mathbf{ijk}) \\
&= \prod_{b\in C(v)\setminus\{c\}} \left(\rho_{b\to v}^{\mathrm{norm}(\mathrm{PTP})}(\mathbf{ij}) + \rho_{b\to v}^{\mathrm{norm}(\mathrm{PTP})}(\mathbf{ik}) + \rho_{b\to v}^{\mathrm{norm}(\mathrm{PTP})}(\mathbf{ijk})\right) \\
&\quad - \prod_{b\in C(v)\setminus\{c\}} \left(\rho_{b\to v}^{\mathrm{norm}(\mathrm{PTP})}(\mathbf{ij}) + \rho_{b\to v}^{\mathrm{norm}(\mathrm{PTP})}(\mathbf{ijk})\right) \\
&\quad - \prod_{b\in C(v)\setminus\{c\}} \left(\rho_{b\to v}^{\mathrm{norm}(\mathrm{PTP})}(\mathbf{ik}) + \rho_{b\to v}^{\mathrm{norm}(\mathrm{PTP})}(\mathbf{ijk})\right) + \prod_{b\in C(v)\setminus\{c\}} \rho_{b\to v}^{\mathrm{norm}(\mathrm{PTP})}(\mathbf{ijk}) \\
&= \lambda_{v\to c}^{(\mathrm{PTP})}(\mathbf{i}).
\end{aligned}$$

Similarly, we can prove that $\lambda_{v\to c}^{(\mathrm{SDBP})}(\mathbf{ij},\mathbf{ijk}) = \lambda_{v\to c}^{(\mathrm{PTP})}(\mathbf{ij})$ and $\lambda_{v\to c}^{(\mathrm{SDBP})}(\mathbf{ijk},\mathbf{ijk}) = \lambda_{v\to c}^{(\mathrm{PTP})}(\mathbf{ijk})$. It then follows that the left correspondence holds.

Now we prove that if the left correspondence holds, then the right correspondence holds. Suppose that the left correspondence holds, then we have

$$\begin{aligned}
\rho_{c\to v}^{\mathrm{norm(PTP)}}(\mathbf{ij}) &= \alpha\cdot\rho_{c\to v}^{(\mathrm{PTP})}(\mathbf{ij}) \\
&= \alpha\cdot\lambda_{V(c)\backslash\{v\}\to c}^{\mathrm{norm(PTP)}}(\mathbf{k}) \\
&= \alpha\left(\beta\cdot\lambda_{V(c)\backslash\{v\}\to c}^{(\mathrm{PTP})}(\mathbf{k})\right) \\
&= \alpha\beta\cdot\lambda_{V(c)\backslash\{v\}\to c}^{(\mathrm{SDBP})}(\mathbf{k},\mathbf{ijk})
\end{aligned}$$

where $\alpha = 1/\sum_{t\in(\chi^*)^v}\rho_{c\to v}^{(\mathrm{PTP})}(t)$ and $\beta = 1/\sum_{t\in(\chi^*)^{V(c)\backslash\{v\}}}\lambda_{V(c)\backslash\{v\}\to c}^{(\mathrm{PTP})}(t)$. We also have

$$\rho_{c\to v}^{*(\mathrm{SDBP})}(\mathbf{ij}) = \delta\cdot\lambda_{V(c)\backslash\{v\}\to c}^{(\mathrm{SDBP})}(\mathbf{k},\mathbf{ijk}).$$

Since both $\rho_{c\to v}^{*(\mathrm{SDBP})}$ and $\rho_{c\to v}^{\mathrm{norm(PTP)}}$ are normalized, it must hold that $\alpha\beta = \delta$. This indicates that $\rho_{c\to v}^{\mathrm{norm(PTP)}}(\mathbf{ij}) = \rho_{c\to v}^{*(\mathrm{SDBP})}(\mathbf{ij})$. Following a similar procedure, one can show that $\rho_{c\to v}^{\mathrm{norm(PTP)}}(\mathbf{ijk}) = \rho_{c\to v}^{*(\mathrm{SDBP})}(\mathbf{ijk})$. This implies that the right correspondence holds.

At this point, we have established the correspondence between passed messages in PTP and those in SDBP.

Now we will prove the summary correspondence (namely, that (5.69) to (5.71)).

$$\begin{aligned}
\mu_v^{(\mathrm{SDBP})}(\mathbf{i}) = &\prod_{c\in C(v)}(\rho_{c\to v}^{*(\mathrm{SDBP})}(\mathbf{ij}) + \rho_{c\to v}^{*(\mathrm{SDBP})}(\mathbf{ik}) + \rho_{c\to v}^{*(\mathrm{SDBP})}(\mathbf{ijk})) \\
&- \prod_{c\in C(v)}(\rho_{c\to v}^{*(\mathrm{SDBP})}(\mathbf{ij}) + \rho_{c\to v}^{*(\mathrm{SDBP})}(\mathbf{ijk})) \\
&- \prod_{c\in C(v)}(\rho_{c\to v}^{*(\mathrm{SDBP})}(\mathbf{ik}) + \rho_{c\to v}^{*(\mathrm{SDBP})}(\mathbf{ijk})) + \prod_{c\in C(v)}\rho_{c\to v}^{*(\mathrm{SDBP})}(\mathbf{ijk})
\end{aligned}$$

$$
\begin{aligned}
= & \prod_{c\in C(v)} \left(\rho_{c\to v}^{\text{norm(PTP)}}(\mathbf{ij}) + \rho_{c\to v}^{\text{norm(PTP)}}(\mathbf{ik}) + \rho_{c\to v}^{\text{norm(PTP)}}(\mathbf{ijk})\right) \\
& - \prod_{c\in C(v)} (\rho_{c\to v}^{\text{norm(PTP)}}(\mathbf{ij}) + \rho_{c\to v}^{\text{norm(PTP)}}(\mathbf{ijk})) \\
& - \prod_{c\in C(v)} (\rho_{c\to v}^{\text{norm(PTP)}}(\mathbf{ik}) + \rho_{c\to v}^{\text{norm(PTP)}}(\mathbf{ijk})) + \prod_{c\in C(v)} \rho_{c\to v}^{\text{norm(PTP)}}(\mathbf{ijk}) \\
= & \ \mu_v^{(\text{PTP})}(\mathbf{i}).
\end{aligned}
$$

Similarly, we can prove that $\mu_v^{(\text{SDBP})}(\mathbf{ij}) = \mu_v^{(\text{PTP})}(\mathbf{ij})$ and $\mu_v^{(\text{SDBP})}(\mathbf{ijk}) = \mu_v^{(\text{PTP})}(\mathbf{ijk})$. This proves the summary correspondence. ■

At this end, it should be convincing that the state-decoupling condition is an important ingredient in the reduction of BP to PTP. It is worth noting that in the case of k-SAT problems, this condition can be imposed simply by the initialization of BP messages. However in the case of 3-COL problems, one needs to manually impose this condition at each iteration, namely, carrying out SDBP instead of BP, so as to arrive at an equivalence to PTP messages. This extra complexity involved in 3-COL problems then suggests that for 3-COL problems, PTP and hence SP are not a special case of BP. Thus at this end, one may conclude that SP is not BP for general CSPs.

Now it remains to investigate, for general CSPs, whether the state-decoupling condition is sufficient for PTP or weighted PTP to reduce from BP, or equivalently *whether* and *when* PTP and weighted PTP are SDBP.

5.4 The Reduction of Weighted PTP from SDBP for General CSPs

Up to this point, we see that the state-decoupling condition critically governs the reduction of BP to PTP (or weighted PTP) for k-SAT problems and 3-COL problems. In this section, we will however show that the state-decoupling condition is not sufficient for BP (more precisely SDBP) to reduce to PTP and that an additional condition is needed in the general context.

Definition 4 (Forceable Token) *For any $(v-c)$, we say that a token $t_v \in (\chi^*)^{\{v\}}$ is* forceable *by Γ_c if there exists a rectangle $\prod_{u\in V(c)\setminus\{v\}} t_u$ on $V(c)\setminus\{v\}$ such that $\mathtt{F}_c\left(\prod_{u\in V(c)\setminus\{v\}} t_u\right) = t_v$.*

We will denote by $\mathcal{F}_c(v)$ the set of all tokens on v that are forceable by Γ_c. Let $\mathcal{A}_c(v) := \bigcup_{t\in\mathcal{F}_c(v)} t$. Since $\mathcal{A}_c(v) = \mathtt{F}_c\left(\prod_{u\in V(c)\setminus\{v\}} (\chi^*)^{\{u\}}\right)$, it follows that $\mathcal{A}_c(v)$ is always forceable. In fact, it is easy to see that $\mathcal{A}_c(v)$ is the "largest" forceable token on v by Γ_c — in the sense of containing all other forceable tokens as its subsets — due to the monotonicity of $\mathtt{F}_c(\cdot)$.

Example 7 *In k-SAT problems, for any $(v-c)$, it is easy to see that $\mathcal{F}_c(v) = \{*, \mathbf{L}\}$, and $\mathcal{A}_c(v) = *$. In 3-COL problems, for any $(v-c)$, it is easy to see that $\mathcal{F}_c(v) = \{\mathbf{123}, \mathbf{12}, \mathbf{23}, \mathbf{13}\}$, and $\mathcal{A}_c(v) = \mathbf{123}$.*

For any $(c-v)$, let $\mathcal{A}_{\sim c}(v)$ be defined by

$$\mathcal{A}_{\sim c}(v) := \bigcap_{b\in C(v)\setminus\{c\}} \mathcal{A}_b(v).$$

Definition 5 (Locally Compatible Constraint) *A constraint Γ_c is said to be locally compatible if for any $v \in V(c)$, any forceable token $t_v \in \mathcal{F}_c(v)$, any rectangle $t' \in \mathtt{F}_c^{-1}(t_v)$ on $V(c)\setminus\{v\}$ (where $\mathtt{F}_c^{-1}(t_v)$ is the set of all rectangles $y_{V(c)\setminus\{v\}}$ on $V(c)\setminus\{v\}$ such that $\mathtt{F}_c(y_{V(c)\setminus\{v\}}) = t_v$) and any $u \in V(c)\setminus\{v\}$, it holds that*

$$\mathcal{A}_{\sim c}(u) \subseteq \mathtt{F}_c\left(t_v \times t'_{:V(c)\setminus\{u,v\}}\right).$$

We note that the local compatibility of a constraint Γ_c as defined above is not simply a property of Γ_c itself. It also relies on the structure of all constraints that are distance-2 away from Γ_c in the factor graph.

Theorem 5 *Let the set of obedience conditionals $\{\omega_v : v \in V\}$ be given, where each $v \in V$ corresponds to a coordinate of a CSP. Let both the MRF of the CSP (that specified via (5.1), (5.2) and (5.3)) and the weighted PTP for the CSP be both parametrized by $\{\omega_v : v \in V\}$.*

Then if every constraint of the CSP is locally compatible, the SDBP derived from the MRF is equivalent to the weighted PTP, where the correspondence is

$$\rho_{c\to v}^{\mathrm{norm(PTP)}} \leftrightarrow \rho_{c\to v}^{*(\mathrm{SDBP})}.$$

Conversely, if such an equivalence holds for every choice of $\{\omega_v : v \in V\}$*, then every constraint of the CSP must be locally compatible.*

Alternatively phrased, Theorem 5 suggests that if the state-decoupling condition is satisfied in every iteration of BP, the local compatibility condition on all constraints is the necessary and sufficient condition for weighted PTP to reduce from BP. — We note that Theorem 5 only refers to the equivalence of right messages. It is however straight-forward to verify (as seen in earlier proofs of equivalent results in this thesis) that right equivalence implies the summary equivalence.

This theorem answers the question *when* SP is SDBP in a general setting.

Proof: Following the message-update rule of SDBP,

$$\begin{aligned}
\rho_{c\to v}^{*(\mathrm{SDBP})}(s_{v,c}^R) \quad &\propto \quad \sum_{s_{V(c)\setminus\{v\},c}^L} \left(\left[s_{v,c}^R = \mathtt{F}_c\left(s_{V(c)\setminus\{v\},c}^L\right)\right] \prod_{u\in V(c)\setminus\{v\}} \lambda_{u\to c}^{(\mathrm{SDBP})}\left(s_{u,c}^L, \mathtt{F}_c\left(s_{V(c)\setminus\{u\},c}^L\right)\right)\right) \\
&= \quad \sum_{s_{V(c)\setminus\{v\},c}^L} \left(\left[s_{v,c}^R = \mathtt{F}_c\left(s_{V(c)\setminus\{v\},c}^L\right)\right] \prod_{u\in V(c)\setminus\{v\}} \sum_{s_{u,C(u)\setminus\{c\}}^R} \right. \\
&\qquad \omega_u\left(s_{u,c}^L \middle| \left(\bigcap_{b\in C(u)\setminus\{c\}} s_{u,b}^R \right) \cap \mathtt{F}_c\left(s_{V(c)\setminus\{u,v\},c}^L \times s_{v,c}^R\right)\right) \\
&\qquad \left. \times \prod_{b\in C(u)\setminus\{c\}} \rho_{b\to u}^{*(\mathrm{SDBP})}(s_{u,b}^R)\right)
\end{aligned} \tag{5.72}$$

Similarly following the message-update rule of weighted PTP, we have

$$\rho_{c\to v}^{\text{norm(PTP)}}(t_{c\to v}) \propto \sum_{t_{V(c)\setminus\{v\}\to c}} \left(\left[t_{c\to v} = \mathtt{F}_c(t_{V(c)\setminus\{v\}\to c})\right] \prod_{u\in V(c)\setminus\{v\}} \sum_{t_{C(u)\setminus\{c\}\to u}} \omega_u \left(t_{u\to c} \middle| \bigcap_{b\in C(u)\setminus\{c\}} t_{b\to u} \right) \cdot \left(\prod_{b\in C(u)\setminus\{c\}} \rho_{b\to u}^{\text{norm(PTP)}}(t_{b\to u}) \right) \right) \quad (5.73)$$

Identifying every right state $s_{v,c}^R$ in (5.72) with token $t_{c\to v}$ in (5.73) and every left state $s_{v,c}^L$ (5.72) with token $t_{v\to c}$ in (5.73), the only difference between (5.72) and (5.73) is the argument of function ω_u. (We note that since both $\rho_{c\to v}^{*(\text{SDBP})}$ and $\rho_{c\to v}^{\text{norm(PTP)}}$ are normalized, the scaling constant in (5.72) and (5.73) are necessarily the same.) We now prove the sufficiency and necessity of the local compatibility condition for the equivalence between $\rho_{c\to v}^{\text{norm(PTP)}}$ and $\rho_{c\to v}^{*(\text{SDBP})}$ via the following chain of two-way implications.

$\rho_{c\to v}^{*(\text{SDBP})} \leftrightarrow \rho_{c\to v}^{\text{norm(PTP)}}, \forall v \in V(c)$

$$\Leftrightarrow \quad \omega_u \left(s_{u,c}^L \middle| \left(\bigcap_{b\in C(u)\setminus\{c\}} s_{u,b}^R \right) \cap \mathtt{F}_c \left(s_{V(c)\setminus\{u,v\},c}^L \times s_{v,c}^R \right) \right) = \omega_u \left(s_{u,c}^L \middle| \bigcap_{b\in C(u)\setminus\{c\}} s_{u,b}^R \right)$$

$\forall v \in V(c)$ and every $\left(s_{v,c}^R, s_{V(c)\setminus\{v\},c}^L\right)$ in the support of $\left[s_{v,c}^R = \mathtt{F}_c\left(s_{V(c)\setminus\{v\},c}^L\right)\right]$,

$\forall u \in V(c)\setminus\{v\}$ and every choice of $|C(u)\setminus\{c\}|$ tokens on $\{u\}, \left\{s_{u,b}^R : b \in C(u)\setminus\{c\}\right\}$,

with each $s_{u,b}^R$ in the support of $\rho_{b\to u}^{(\text{PTP})}$.

$$\Leftrightarrow \quad \left(\bigcap_{b\in C(u)\setminus\{c\}} s_{u,b}^R \right) \cap \mathtt{F}_c \left(s_{V(c)\setminus\{u,v\},c}^L \times s_{v,c}^R \right) = \bigcap_{b\in C(u)\setminus\{c\}} s_{u,b}^R$$

$\forall v \in V(c)$ and every $\left(s_{v,c}^R, s_{V(c)\setminus\{v\},c}^L\right)$ such that $s_{v,c}^R \in \mathcal{F}_c(v)$ and $s_{V(c)\setminus\{v\},c}^L \in \mathtt{F}_c^{-1}(s_{v,c}^R)$,

$\forall u \in V(c)\setminus\{v\}$ and every choice of $|C(u)\setminus\{c\}|$ tokens on $\{u\}, \left\{s_{u,b}^R : b \in C(u)\setminus\{c\}\right\}$,

with each $s_{u,b}^R \in \mathcal{F}_b(u)$.

$$\Leftrightarrow \quad \bigcap_{b\in C(u)\setminus\{c\}} s_{u,b}^R \subseteq \mathtt{F}_c \left(s_{V(c)\setminus\{u,v\},c}^L \times s_{v,c}^R \right)$$

$\forall v \in V(c)$ and every $\left(s_{v,c}^R, s_{V(c)\setminus\{v\},c}^L\right)$ such that $s_{v,c}^R \in \mathcal{F}_c(v)$ and $s_{V(c)\setminus\{v\},c}^L \in \mathtt{F}_c^{-1}(s_{v,c}^R)$,

$\forall u \in V(c)\setminus\{v\}$ and every choice of $|C(u)\setminus\{c\}|$ tokens on $\{u\}, \left\{s_{u,b}^R : b \in C(u)\setminus\{c\}\right\}$,

with each $s_{u,b}^R \in \mathcal{F}_b(u)$.

$\Leftrightarrow \quad \bigcap_{b \in C(u) \setminus \{c\}} \mathcal{A}_b(u) \subseteq \mathtt{F}_c \left(s^L_{V(c)\setminus\{u,v\},c} \times s^R_{v,c} \right)$

$\forall v \in V(c)$ and every $\left(s^R_{v,c}, s^L_{V(c)\setminus\{v\},c} \right)$ such that $s^R_{v,c} \in \mathcal{F}_c(v)$ and $s^L_{V(c)\setminus\{v\},c} \in \mathtt{F}_c^{-1}(s^R_{v,c})$, and every $u \in V(c) \setminus \{v\}$.

$\Leftrightarrow \quad \mathcal{A}_{\sim c}(u) \subseteq \mathtt{F}_c \left(s^L_{V(c)\setminus\{u,v\},c} \times s^R_{v,c} \right)$,

$\forall v \in V(c)$ and every $\left(s^R_{v,c}, s^L_{V(c)\setminus\{v\},c} \right)$ such that $s^R_{v,c} \in \mathcal{F}_c(v)$ and $s^L_{V(c)\setminus\{v\},c} \in \mathtt{F}_c^{-1}(s^R_{v,c})$, and every $u \in V(c) \setminus \{v\}$.

$\Leftrightarrow \quad$ Constraint Γ_c is locally compatible.

Thus

$$\rho^{\text{norm(PTP)}}_{c \to v} \leftrightarrow \rho^{*\text{(SDBP)}}_{c \to v}, \text{ for every } (x_v, \Gamma_c) \in E(G)$$

$\Leftrightarrow \quad$ Every constraint Γ_c is locally compatible.

■

Now it is easy to verify that for both k-SAT and 3-COL problems, the fact that PTP or weighted PTP can be reduced from BP with state-decoupling condition imposed is due to the fact that every constraint is locally compatible.

For k-SAT problems, as noted earlier, $\mathcal{F}_c(v) = \{\mathbf{L}, *\}$. If we pick t_v to be either token from $\mathcal{F}_c(v)$, then for any $t' \in \mathtt{F}_c^{-1}(t_v)$ and any $u \in V(c) \setminus \{v\}$, it can be verified that $\mathtt{F}_c \left(t'_{:V(c)\setminus\{u,v\}} \times t_v \right) = *$. This makes $\mathcal{A}_{\sim c}(u) \subseteq \mathtt{F}_c \left(t'_{:V(c)\setminus\{u,v\}} \times t_v \right)$ always satisfied, independent of the factor graph structure of the problem instance.

For 3-COL problems, as noted earlier, we see $\mathcal{F}_c(v) = \{\mathbf{123}, \mathbf{12}, \mathbf{23}, \mathbf{13}\}$. Suppose that u is the only other coordinate (except v) that is involved in constraint Γ_c. If we pick t_v to be any token from $\mathcal{F}_c(v)$, then $\mathtt{F}^u_c(t_v) = \mathbf{123}$. This again makes $\mathcal{A}_{\sim c}(u) \subseteq \mathtt{F}^u_c(t_v)$ always satisfied, independent of the factor graph structure of the problem instance.

That is, in both k-SAT and 3-COL problems, the structure of each local constraint *alone* guarantees the local compatibility condition satisfied by every constraint, irrespective of how a constraint interacts with other constraints (that are distance 2 apart) as is generally required in the local compatibility condition. We generalize this fact in the following corollary — immediately following Theorem 5 — which provides a sufficient condition for SDBP to reduce to PTP without relying on the interaction of neighboring constraints. For CSPs constructed with generic local constraint by random factor graph structure, the corollary may turn out to be useful.

Corollary 2 *Let both the MRF of the CSP (specified via (5.1), (5.2) and (5.3)) and the weighted PTP for the CSP be parametrized by the same* $\{\omega_v : v \in V\}$. *Suppose that every constraint* Γ_c *is such that for any* $v \in V(c)$, *any forceable token* $t_v \in \mathcal{F}_c(v)$, *any rectangle* $t' \in \mathtt{F}_c^{-1}(t_v)$ *on* $V(c) \setminus \{v\}$, *and any* $u \in V(c) \setminus \{v\}$, *it holds that*

$$\mathtt{F}_c\left(t_v \times t'_{:V(c)\setminus\{u,v\}}\right) = (\chi^*)^{\{v\}}.$$

Then SDBP derived from the MRF is equivalent to weighted PTP, where the correspondence is

$$\rho_{c\to v}^{\mathrm{norm(PTP)}} \leftrightarrow \rho_{c\to v}^{*\mathrm{(SDBP)}}.$$

For completeness, we conclude this section by constructing an example of CSP in which the local compatibility condition is not satisfied by every constraint.

Example 8 *Suppose that* Γ_c *and* Γ_b *are two of the constraints defining a CSP, and the factor graph representing the CSP locally obeys the structure shown in Figure 5.2. Suppose that each variable of the CSP has alphabet* $\chi = \{0, 1, 2\}$ *and that* Γ_c *is defined as* $\Gamma_c := \{(0_v, 0_u), (0_v, 1_u), (1_v, 2_u), (2_v, 2_u)\}$. *Suppose that* Γ_b *is defined as* $\Gamma_b := \{(0_u, 0_w), (1_u, 1_w), (2_u, 1_w)\}$. *Note that* $\mathcal{F}_c(v) = \{\mathbf{0}_v, \mathbf{12}_v, \mathbf{012}_v\}$, *and it is easy to verify that* $\mathcal{A}_{\sim c}(u) = \mathcal{A}_b(u) = \mathtt{F}_b(\mathbf{012}_w) = \mathbf{012}_u$. *Now if we pick* $t_v = \mathbf{0}_v$, *then we have* $\mathcal{A}_{\sim c}(u) \not\subseteq \mathtt{F}_c(t_v) = \mathbf{01}_u$. *Thus constraint* Γ_c *is not*

locally compatible, and following Theorem 5, PTP or weighted PTP can not be reduced from SDBP for this CSP.

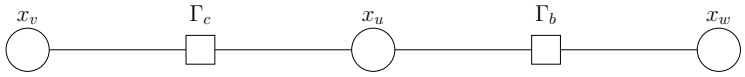


Figure 5.2: A portion of a factor graph G.

With this example, we see that it is not always the case that SDBP is SP.

5.5 Concluding Remarks

In this chapter, we present a normally realized MRF formalism for arbitrary CSPs, in the style of [23]. We derive the BP message-update equations from such an MRF, which we use as the basis for answering the question "is SP BP?" in the general context. We show that although it is possible, in the context of k-SAT problems, to reduce the SP (or weighted or non-weighted PTP) message-update rule from these BP equations, the claim that "SP is BP" is false in general. This clarifies the earlier misconception that understands SP as a special case of BP. We also establish necessary and sufficient conditions under which one may validly regard SP as an instance of BP.

A consequence of this result is the possibility that new analytic tools may be needed for analyzing SP algorithms, since direct applications of the existing tools for BP are likely to be less effective. Some machineries from statistical physics seem particularly promising and for a comprehensive review of the related subjects therein, the reader is referred to [25], a work being developed concurrently with this thesis.

Chapter 6

Graph Coloring with Weighted PTP

In this chapter, we apply the generalized survey propagation algorithm, namely, the weighted PTP algorithm, to 3-COL problems on Erdös-Renyi random graphs. It is known that this is a family of NP-complete problems and an SP-based solver was recently proposed by Braunstein et al [10]. We will first describe the problem and then propose a weighted PTP based algorithm to investigate whether there may be benefit resulting from this generalized notion of SP. Experimental results are then presented and the performance of our algorithm is compared with the algorithm of [10] (which we refer to as the BMPWZ algorithm). The chapter is concluded with a brief discussion.

6.1 The 3-COL Problem on Erdös-Renyi Random Graphs

Recall as introduced in Section 3.1.2, given an undirected graph (Δ, Ξ) with vertex set Δ and edge set Ξ, the objective of the 3-COL problem on (Δ, Ξ) is to assign each vertex in Δ a color from 3 different colors (each denoted by a number in set $\chi = \{1, 2, 3\}$) such that every pair of adjacent vertices have different colors. Formulated as a CSP on a factor graph, the graph

vertices Δ may be identified with the coordinate set V and the graph edges Ξ may be identified with constraint-indexing set C. In particular, each constraint connects precisely two variables in the factor graph.

We now consider the 3-COL problems on Erdös-Renyi random graphs. For any integer $n > 1$ and any real number $p \in (0, 1)$, the Erdös-Renyi random graph ensemble $\mathcal{G}_n(p)$ is the ensemble of random graphs each containing n vertices and in which every pair of vertices are connected by an edge independently generated with probability p. It is straight-forward to verify that the expected total number of edges in a random graph drawn from $\mathcal{G}_n(p)$ is $\binom{n}{2}p$, and the expected (or average) vertex degree of any vertex in the graph, which we denote by θ, is

$$\theta = 2 \times \binom{n}{2} p/n = (n-1)p.$$

As we typically consider large random graphs (i.e., $n \gg 1$), we see that $\theta \approx np$. Instead of using p, the Erdös-Renyi random graph ensemble $\mathcal{G}_n(p)$ may be alternatively parameterized by θ, in which case we denote the ensemble by $\mathcal{G}_n[\theta]$.

It is well-known that 3-COL problems are in general NP-complete. On Erdös-Renyi random graph ensembles, one expects a phase transition phenomenon in the hardness of the 3-COL problems. Similar to the random k-SAT problems, it is shown that there are two thresholds of parameter θ: $\theta_c \approx 4.42$ and $\theta_d \approx 4.69$; when $\theta < \theta_c$, 3-COL on $\mathcal{G}_n[\theta]$ is easy and solvable by local search algorithms; when $\theta > \theta_d$, the problem does not have solution almost surely; when θ is between the two thresholds, the problem is hard and local search algorithms almost always fail. In the hard regime of 3-COL on $\mathcal{G}_n[\theta]$, Braunstein et al. applied a survey propagation based algorithm, which we call the BMPWZ algorithm, to solve the problems [10]. The performance of their algorithm is shown in Figure 6.1.

As shown in Figure 6.1, the BMPWZ algorithm is capable of solving a significant range of the problems in the hard regime. However, there is still a range of θ in which the algorithm fails.

In this study, we will apply weighted PTP to 3-COL on $\mathcal{G}_n[\theta]$ in the hard regime (namely,

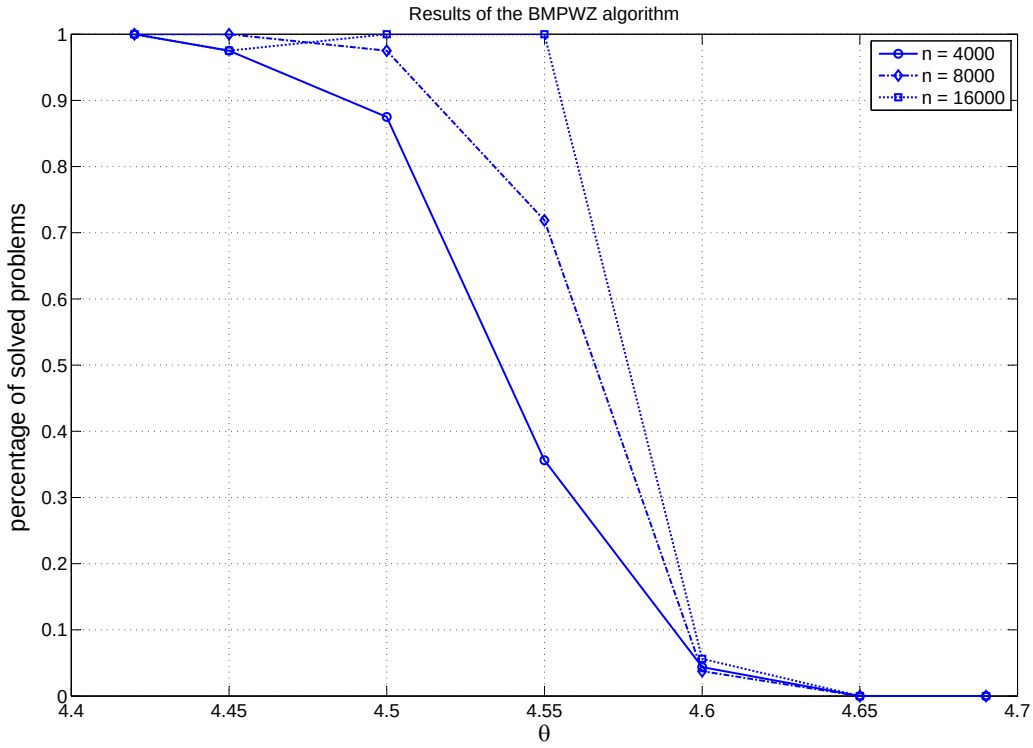


Figure 6.1: The fraction of 3-COL problems on $\mathcal{G}_n[\theta]$ solved by the BMPWZ algorithm. The plots are reproduced by visual inspection of the plots in [10].

$\theta \in (4.42, 4.69)$) and investigate whether this generalized notion of SP may potentially push further the solvability of such problems.

6.2 Solving 3-COL Problems by Weighted PTP with Decimation

As in their applications to other problems, SP-based algorithms typically involve

- a local message-update rule for variables, for constraints, or for both;
- a set of decimation rules that select a subset of the variables and fixes their values so as to reduce the size of the problem;

- a schedule that orders and iterates over message-update procedures and decimation procedures; and
- a local-search algorithm that assigns the values of the remaining variables when the size of the problem is relatively small.

We now propose an algorithm based on the generalized SP, or weighted PTP, to answer the question if there is any performance benefit by using the generalized SP. Although the key distinction of our algorithm from the BMPWZ algorithm in [10] is on the message-update rule, there are also several other differences, as are listed below.

- Instead of using a complicated decimation rule as in the BMPWZ algorithm (involving seven different cases), we use two very simple decimation rules, which we call "gentle decimation" and "aggressive decimation".
- The schedule of the proposed algorithm has three nested loops, called "iterations", "rounds", and "epoches".
- The local-search algorithm is eliminated for the simplicity of implementation.

We first describe the message-update and decimation rules, and then the schedule of the algorithm.

6.2.1 Message-Update Rule

It is clear that specifying the message-update rule in weighted PTP boils down to specifying function ω_v for each $v \in V$. Since each vertex v in $\mathcal{G}_n[\theta]$ is statistically identical and the three colors are symmetric, the most natural choice of $\{\omega_v : v \in V\}$ is to make ω_v independent of v and parameterize it by three real numbers $\alpha, \beta, \gamma \in [0, 1]$ as follows.

$$\omega_v(a|b) := \begin{cases} 1, & \text{if } a = b = \mathbf{i} \\ \frac{\alpha}{2}, & \text{if } a \subset b = \mathbf{ij} \\ 1-\alpha, & \text{if } a = b = \mathbf{ij} \\ \frac{\beta}{3}, & \text{if } a = \mathbf{i} \text{ and } b = \mathbf{ijk} \\ \frac{\gamma}{3}, & \text{if } a = \mathbf{ij} \text{ and } b = \mathbf{ijk} \\ 1-\beta-\gamma, & \text{if } a = b = \mathbf{ijk} \\ 0, & \text{otherwise.} \end{cases} \tag{6.1}$$

Using this choice of ω_v, it is straightforward to verify that the weighted PTP message-update rules (equations (4.33) to (4.35)) reduces to equations shown in Figures 6.2 to 6.4.

We note that Figures 6.2 to 6.4, λ^{norm} is the normalized left message, namely, the scaled version of function λ such that the sum of the function over its support equals 1. Similarly, ρ^{norm} is the normalized right message, defined in a similar manner.

6.2.2 Decimation Rules

There are two kinds of decimation procedures that will be applied, "gentle decimation" and "aggressive decimation", the rules of which we now explain.

- **Gentle Decimation** Gentle decimation is parameterized by two threshold values in $(0, 1)$, T_{fix} and T_{forbid}, where T_{fix} is relatively close to 1 and T_{forbid} is relatively close to 0. The gentle decimation procedure takes as the input the summary messages at all variables (that have not been fixed to a color) and, if successful, *fixes* one variable to a particular color or *forbids* a variable to take a particular color. More precisely, the procedure first attempts to find a variable whose summary message is the most biased to one singleton token and if the bias is greater than T_{fix}, it fixes the variable to the color corresponding to the singleton token; if this is not possible, the procedure looks for a variable whose bias on some singleton token is the smallest across all variables and all colors, subject to the constraint that the bias on this singleton is below the threshold

3-COL Left Message-Update Rule

$$
\begin{aligned}
\lambda_{v\to c}(\mathbf{1}) := & \prod_{b\in C(v)\setminus\{c\}} (\rho^{\text{norm}}_{b\to v}(\mathbf{12}) + \rho^{\text{norm}}_{b\to v}(\mathbf{13}) + \rho^{\text{norm}}_{b\to v}(\mathbf{123})) \\
- & \left(1-\frac{\alpha}{2}\right)\cdot \prod_{b\in C(v)\setminus\{c\}} (\rho^{\text{norm}}_{b\to v}(\mathbf{12}) + \rho^{\text{norm}}_{b\to v}(\mathbf{123})) \\
- & \left(1-\frac{\alpha}{2}\right)\cdot \prod_{b\in C(v)\setminus\{c\}} (\rho^{\text{norm}}_{b\to v}(\mathbf{13}) + \rho^{\text{norm}}_{b\to v}(\mathbf{123})) \\
+ & \left(1-\alpha+\frac{\beta}{3}\right)\cdot \prod_{b\in C(v)\setminus\{c\}} \rho^{\text{norm}}_{b\to v}(\mathbf{123}) \\
\lambda_{v\to c}(\mathbf{2}) := & \prod_{b\in C(v)\setminus\{c\}} (\rho^{\text{norm}}_{b\to v}(\mathbf{12}) + \rho^{\text{norm}}_{b\to v}(\mathbf{23}) + \rho^{\text{norm}}_{b\to v}(\mathbf{123})) \\
- & \left(1-\frac{\alpha}{2}\right)\cdot \prod_{b\in C(v)\setminus\{c\}} (\rho^{\text{norm}}_{b\to v}(\mathbf{12}) + \rho^{\text{norm}}_{b\to v}(\mathbf{123})) \\
- & \left(1-\frac{\alpha}{2}\right)\cdot \prod_{b\in C(v)\setminus\{c\}} (\rho^{\text{norm}}_{b\to v}(\mathbf{23}) + \rho^{\text{norm}}_{b\to v}(\mathbf{123})) \\
+ & \left(1-\alpha+\frac{\beta}{3}\right)\cdot \prod_{b\in C(v)\setminus\{c\}} \rho^{\text{norm}}_{b\to v}(\mathbf{123}) \\
\lambda_{v\to c}(\mathbf{3}) := & \prod_{b\in C(v)\setminus\{c\}} (\rho^{\text{norm}}_{b\to v}(\mathbf{13}) + \rho^{\text{norm}}_{b\to v}(\mathbf{23}) + \rho^{\text{norm}}_{b\to v}(\mathbf{123})) \\
- & \left(1-\frac{\alpha}{2}\right)\cdot \prod_{b\in C(v)\setminus\{c\}} (\rho^{\text{norm}}_{b\to v}(\mathbf{13}) + \rho^{\text{norm}}_{b\to v}(\mathbf{123})) \\
- & \left(1-\frac{\alpha}{2}\right)\cdot \prod_{b\in C(v)\setminus\{c\}} (\rho^{\text{norm}}_{b\to v}(\mathbf{23}) + \rho^{\text{norm}}_{b\to v}(\mathbf{123})) \\
+ & \left(1-\alpha+\frac{\beta}{3}\right)\cdot \prod_{b\in C(v)\setminus\{c\}} \rho^{\text{norm}}_{b\to v}(\mathbf{123}) \\
\lambda_{v\to c}(\mathbf{12}) := & (1-\alpha)\cdot \prod_{b\in C(v)\setminus\{c\}} (\rho^{\text{norm}}_{b\to v}(\mathbf{12}) + \rho^{\text{norm}}_{b\to v}(\mathbf{123})) - \left(1-\alpha-\frac{\gamma}{3}\right)\cdot \prod_{b\in C(v)\setminus\{c\}} \rho^{\text{norm}}_{b\to v}(\mathbf{123}) \\
\lambda_{v\to c}(\mathbf{13}) := & (1-\alpha)\cdot \prod_{b\in C(v)\setminus\{c\}} (\rho^{\text{norm}}_{b\to v}(\mathbf{13}) + \rho^{\text{norm}}_{b\to v}(\mathbf{123})) - \left(1-\alpha-\frac{\gamma}{3}\right)\cdot \prod_{b\in C(v)\setminus\{c\}} \rho^{\text{norm}}_{b\to v}(\mathbf{123}) \\
\lambda_{v\to c}(\mathbf{23}) := & (1-\alpha)\cdot \prod_{b\in C(v)\setminus\{c\}} (\rho^{\text{norm}}_{b\to v}(\mathbf{23}) + \rho^{\text{norm}}_{b\to v}(\mathbf{123})) - \left(1-\alpha-\frac{\gamma}{3}\right)\cdot \prod_{b\in C(v)\setminus\{c\}} \rho^{\text{norm}}_{b\to v}(\mathbf{123}) \\
\lambda_{v\to c}(\mathbf{123}) := & (1-\beta-\gamma)\cdot \prod_{b\in C(v)\setminus\{c\}} \rho^{\text{norm}}_{b\to v}(\mathbf{123})
\end{aligned}
$$

Figure 6.2: Proposed algorithm for 3-COL: left message update rule

3-COL Right Message-Update Rule

$$
\begin{aligned}
\rho_{c\to v}(\mathbf{12}) &:= \lambda^{\text{norm}}_{u\to c}(\mathbf{3}) \\
\rho_{c\to v}(\mathbf{13}) &:= \lambda^{\text{norm}}_{u\to c}(\mathbf{2}) \\
\rho_{c\to v}(\mathbf{23}) &:= \lambda^{\text{norm}}_{u\to c}(\mathbf{1}) \\
\rho_{c\to v}(\mathbf{123}) &:= \lambda^{\text{norm}}_{u\to c}(\mathbf{12}) + \lambda^{\text{norm}}_{u\to c}(\mathbf{13}) + \lambda^{\text{norm}}_{u\to c}(\mathbf{23}) + \lambda^{\text{norm}}_{u\to c}(\mathbf{123})
\end{aligned}
$$

Figure 6.3: Proposed algorithm for 3-COL: right message update rule

T_{forbid} and that the biases for this variable on other singleton tokens are not lower than T_{forbid}. If the procedure is able to fix or forbid, it returns `true`, indicating successful decimation; otherwise, it returns `false`, indicating a decimation failure.

- **Aggressive Decimation** Aggressive decimation, also taking the summary messages at all unfixed variables as input, simply finds the variable whose summary message is most biased to one singleton token and fixes the variable to the corresponding color.

We note that if a variable is fixed to a color or forbidden to take a particular color, in the subsequent updates of its left messages, certain modification of the left messages is required. We now give the specifics of such modification.

- If a variable is fixed to a particular color, say $i \in \{1, 2, 3\}$, its left message is always the probability mass function that puts probability 1 on the singleton token corresponding to that color.

- If a variable is forbidden to take a particular color, say color $i \in \{1, 2, 3\}$, the left message λ, after being updated according to the regular update rule, is modified to $\tilde{\lambda}$ by setting

$$
\begin{aligned}
\tilde{\lambda}(\mathbf{i}) &= \tilde{\lambda}(\mathbf{ij}) = \tilde{\lambda}(\mathbf{ik}) = \tilde{\lambda}(\mathbf{ijk}) = 0 \\
\tilde{\lambda}(\mathbf{j}) &= \lambda(\mathbf{ij}) + \lambda(\mathbf{j}) \\
\tilde{\lambda}(\mathbf{k}) &= \lambda(\mathbf{ik}) + \lambda(\mathbf{k}) \\
\tilde{\lambda}(\mathbf{jk}) &= \lambda(\mathbf{ijk}) \qquad (6.2)
\end{aligned}
$$

3-COL Summary Message-Update Rule

$$
\begin{aligned}
\mu_v(\mathbf{1}) := & \prod_{b\in C(v)} (\rho^{\text{norm}}_{b\to v}(\mathbf{12}) + \rho^{\text{norm}}_{b\to v}(\mathbf{13}) + \rho^{\text{norm}}_{b\to v}(\mathbf{123})) \\
& - \left(1-\frac{\alpha}{2}\right)\cdot \prod_{b\in C(v)} (\rho^{\text{norm}}_{b\to v}(\mathbf{12}) + \rho^{\text{norm}}_{b\to v}(\mathbf{123})) \\
& - \left(1-\frac{\alpha}{2}\right)\cdot \prod_{b\in C(v)} (\rho^{\text{norm}}_{b\to v}(\mathbf{13}) + \rho^{\text{norm}}_{b\to v}(\mathbf{123})) + \left(1-\alpha+\frac{\beta}{3}\right)\cdot \prod_{b\in C(v)} \rho^{\text{norm}}_{b\to v}(\mathbf{123}) \\
\mu_v(\mathbf{2}) := & \prod_{b\in C(v)} (\rho^{\text{norm}}_{b\to v}(\mathbf{12}) + \rho^{\text{norm}}_{b\to v}(\mathbf{23}) + \rho^{\text{norm}}_{b\to v}(\mathbf{123})) \\
& - \left(1-\frac{\alpha}{2}\right)\cdot \prod_{b\in C(v)} (\rho^{\text{norm}}_{b\to v}(\mathbf{12}) + \rho^{\text{norm}}_{b\to v}(\mathbf{123})) \\
& - \left(1-\frac{\alpha}{2}\right)\cdot \prod_{b\in C(v)} (\rho^{\text{norm}}_{b\to v}(\mathbf{23}) + \rho^{\text{norm}}_{b\to v}(\mathbf{123})) + \left(1-\alpha+\frac{\beta}{3}\right)\cdot \prod_{b\in C(v)} \rho^{\text{norm}}_{b\to v}(\mathbf{123}) \\
\mu_v(\mathbf{3}) := & \prod_{b\in C(v)} (\rho^{\text{norm}}_{b\to v}(\mathbf{13}) + \rho^{\text{norm}}_{b\to v}(\mathbf{23}) + \rho^{\text{norm}}_{b\to v}(\mathbf{123})) \\
& - \left(1-\frac{\alpha}{2}\right)\cdot \prod_{b\in C(v)} (\rho^{\text{norm}}_{b\to v}(\mathbf{13}) + \rho^{\text{norm}}_{b\to v}(\mathbf{123})) \\
& - \left(1-\frac{\alpha}{2}\right)\cdot \prod_{b\in C(v)} (\rho^{\text{norm}}_{b\to v}(\mathbf{23}) + \rho^{\text{norm}}_{b\to v}(\mathbf{123})) + \left(1-\alpha+\frac{\beta}{3}\right)\cdot \prod_{b\in C(v)} \rho^{\text{norm}}_{b\to v}(\mathbf{123}) \\
\mu_v(\mathbf{12}) := & (1-\alpha)\cdot \prod_{b\in C(v)} (\rho^{\text{norm}}_{b\to v}(\mathbf{12}) + \rho^{\text{norm}}_{b\to v}(\mathbf{123})) - \left(1-\alpha-\frac{\gamma}{3}\right)\cdot \prod_{b\in C(v)} \rho^{\text{norm}}_{b\to v}(\mathbf{123}) \\
\mu_v(\mathbf{13}) := & (1-\alpha)\cdot \prod_{b\in C(v)} (\rho^{\text{norm}}_{b\to v}(\mathbf{13}) + \rho^{\text{norm}}_{b\to v}(\mathbf{123})) - \left(1-\alpha-\frac{\gamma}{3}\right)\cdot \prod_{b\in C(v)} \rho^{\text{norm}}_{b\to v}(\mathbf{123}) \\
\mu_v(\mathbf{23}) := & (1-\alpha)\cdot \prod_{b\in C(v)} (\rho^{\text{norm}}_{b\to v}(\mathbf{23}) + \rho^{\text{norm}}_{b\to v}(\mathbf{123})) - \left(1-\alpha-\frac{\gamma}{3}\right)\cdot \prod_{b\in C(v)} \rho^{\text{norm}}_{b\to v}(\mathbf{123}) \\
\mu_v(\mathbf{123}) := & (1-\beta-\gamma)\cdot \prod_{b\in C(v)} \rho^{\text{norm}}_{b\to v}(\mathbf{123})
\end{aligned}
$$

Figure 6.4: Proposed algorithm for 3-COL: summary message update rule

Then normalization may be performed as needed.

We note that if color i is forbidden, the *support* of the message is no longer the entire extended alphabet, but rather the set $\{\mathbf{j}, \mathbf{k}, \mathbf{jk}\}$.

6.2.3 Message-Update and Decimation Schedule

The update of messages and decimation procedures are packaged in three nested levels of loop, which we call iterations, rounds, and epoches. Each epoch encapsulates many rounds, and each round encapsulates many iterations.

The global schedule of updating messages and executing decimation is summarized in the pseudo-code in Figure 6.5, which we now explain in detail.

- Throughout the algorithm, the left message of each degree-one variable always passes the distribution that puts probability mass 1 on token * or **ijk**.

- The innermost loops are called iterations.

  In each iteration, all constraints update their right messages followed by all variables updating their left messages.

- A round encapsulates iterations.

  Before the loop over iterations in each round, all left messages are initialized. If a variable has not been fixed, we initialize its left message randomly. That is, for each token in the support of the message, we assign a random number uniformly distributed in the interval $(0, 1)$ and normalize the message so that it is a distribution (on its support).

  In each round, iterations are terminated when the messages converge or when a predetermined number, M_{it}, of iterations is reached. Then all summary messages are computed and a gentle decimation is performed.

- An epoch encapsulates rounds.

Pseudo-Code of Proposed Algorithm

```
EPOCH=1;
do //loop over EPOCHES
{
    if (EPOCH==1)
        fix v* and v**;
    else
        aggressiveDecimation();


    ROUND=1;
    do //loop over ROUNDS
    {
        initializeLeftMessages();
        ITERATION=1;
        while(messagesHaveNotConverged or ITERATION<M_it)
        //loop over ITERATIONS
        {
            updateAllRightMessages();
            updateAllLeftMessages();
            ITERATION ++;
        }
        updateAllSummaryMessages();
        isGentleDecimationSucessful=gentleDecimation();
        ROUND ++;
    }while(isGentleDecimationSuccessful)
    EPOCH ++;
}while(problemSolved or conflictDetected)
```

Figure 6.5: Pseudo-Code of Proposed Coloring Algorithm

Except for the first epoch, every epoch begins with an aggressive decimation. The first epoch begins with fixing two variables, namely those indexed by v^* and v^{**}; here v^* is the vertex (in the graph to be colored) that has the maximum degree, and v^{**} is the neighboring vertex of v^* (in the graph to be colored) having the maximum degree. The

variable indexed by v^* is fixed to an arbitrary color, say 1, and the variable indexed by v^{**} is fixed to a different color, say 2.

In each epoch, the loop over rounds is terminated when the gentle decimation fails, namely, when no variable is fixed or forbidden.

- The loop over epoches is terminated when either one of the following two conditions holds:
  - Each variable has been fixed to a color and there is no conflict between the assigned colors of neighboring variables, in which case, the problem is solved.
  - Conflict occurs between the assigned colors of some neighboring variables, in which case the algorithm declares a failure.

6.3 Experimental Results

6.3.1 Parameter Setting

The performance of our coloring algorithm is clearly related to the parameter setting of the algorithm, particularly (α, β, γ). We first performed an investigation on how to choose these parameters.

Under various trial parameter settings, we observed that when the algorithm fails to solve a problem, in most of the cases, it fails in the first epoch. Based on this observation, we used various parameter settings of (α, β, γ) and ran the algorithm on 50 random graphs from ensemble $\mathcal{G}_{4000}[4.55]$ up to the end of the first epoch. The frequency at which the algorithm passed the first epoch is plotted in Figure 6.6.

The results plotted in Figure 6.6 suggest that the best parameter setting for (α, β, γ) is near $(0.09375, 0.0625, 0.25)$.

Based on this investigation, we set parameters (α, β, γ) to the above configuration. Other parameters appear less critical, which are set as follows: $T_{\text{fix}} = 0.9$, $T_{\text{forbid}} = 0.15$, $M_{\text{it}} = 400$.

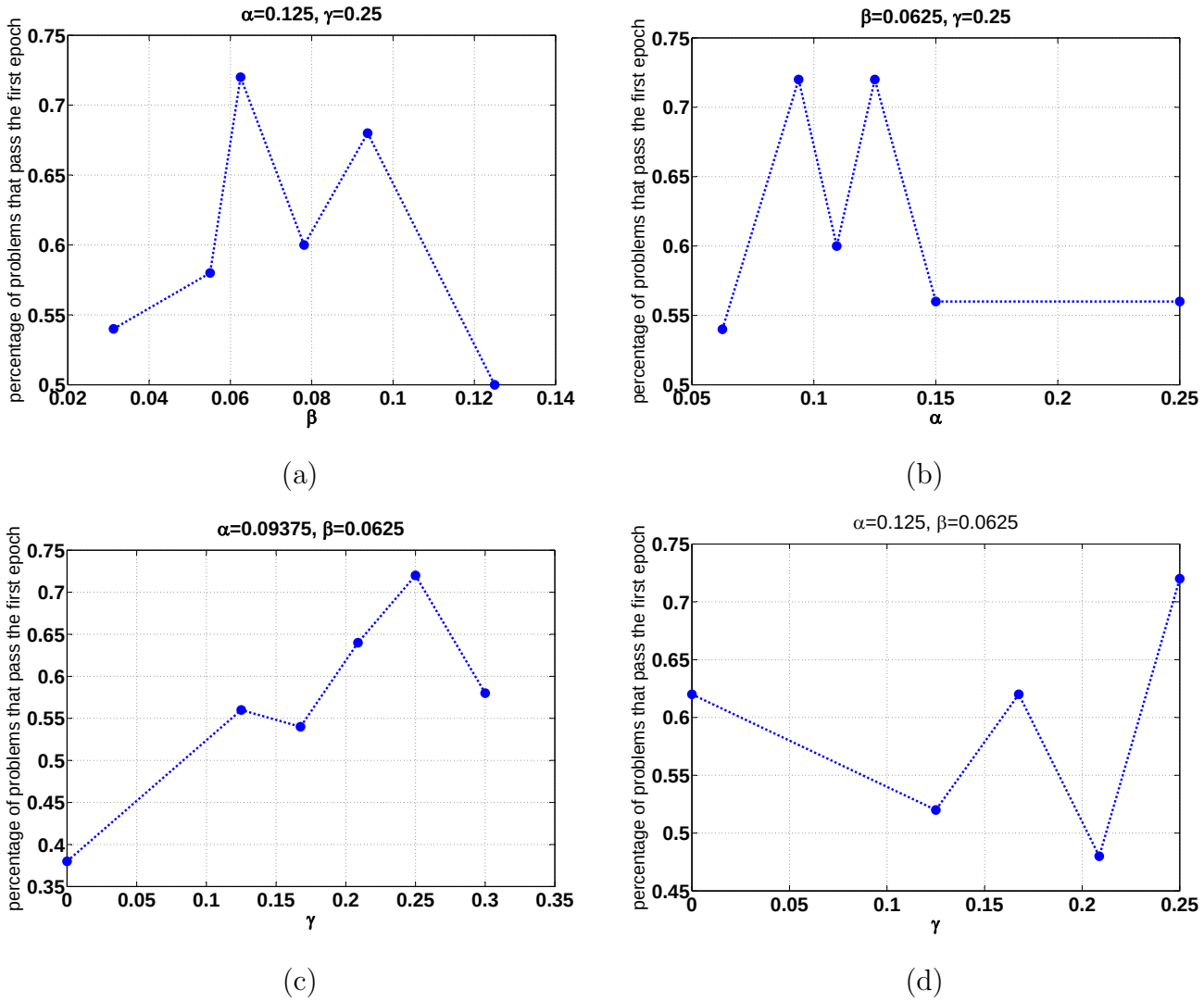


Figure 6.6: The frequency at which the algorithm passed the first epoch for various settings of (α, β, γ).

6.3.2 Behavior of Proposed Algorithm

We ran the algorithm on $\mathcal{G}_n[\theta]$ for $n = 4000, 8000, 16000$ in the hard regime of θ ($\theta \in [4.40, 4.70]$), where from each graph ensemble, 50 graphs are drawn.

We note that since aggressive decimation fixes the most biased variable without taking into account the extent of bias, the aggressive decimation procedures impact the behavior of

proposed algorithm significantly. In particular, because aggressive decimation starts from the second epoch, we investigated how the algorithm behaves in the first epoch and in the later epoches, respectively.

Figure 6.7 plots the average fraction of variables that are fixed in the first epoch across all solved problems, against θ and for each choice of n. Although when the average vertex degree approaches the algorithm's solvability limit, the three curves behaves somewhat differently, within the limit, consistency among the three curves is observed. Specifically, the algorithm consistently fixes about 40% to 60% of the variables on average. In addition, the fraction of fixed variables increases with average vertex degree θ; this trend is expected, since as the average vertex degree increases, fixing a variable will immediately impact a correspondingly increased number of other vertices (namely, the adjacent vertices) in the graph.

On the other hand, it is also revealed from Figure 6.7 that the first epoch does not reduce the problem to a sufficiently small size.

Figures 6.8 to 6.10 each plot the average and the maximum number of variables fixed in each later epoch (after the first epoch). Deduced from these figures, most later epoches fixes only one variable, although there are sporadic epoches which may fix much more. This suggests that the algorithm after the first epoch behaves like a crude "local search" algorithm: In an epoch when a relatively large number of variables are fixed, the local search is more effective and finds a relatively long "path" of variables to fix. In the majority of later epoches, however, only one variable (namely, the one fixed by aggressive decimation) is fixed in each epoch; in this case, the algorithm is prone to fixing a wrong variable.

The observations from Figures 6.7 to 6.10 suggest that the decimation rules used in the proposed algorithm (under the global schedule) are rather far from being optimal. That is, the algorithm seems to have started its "local search" too early and the local search is not done in the most effective manner. Even though this is the case, we will show next that the proposed algorithm does bring some advantages comparing with the BMPWZ algorithm.

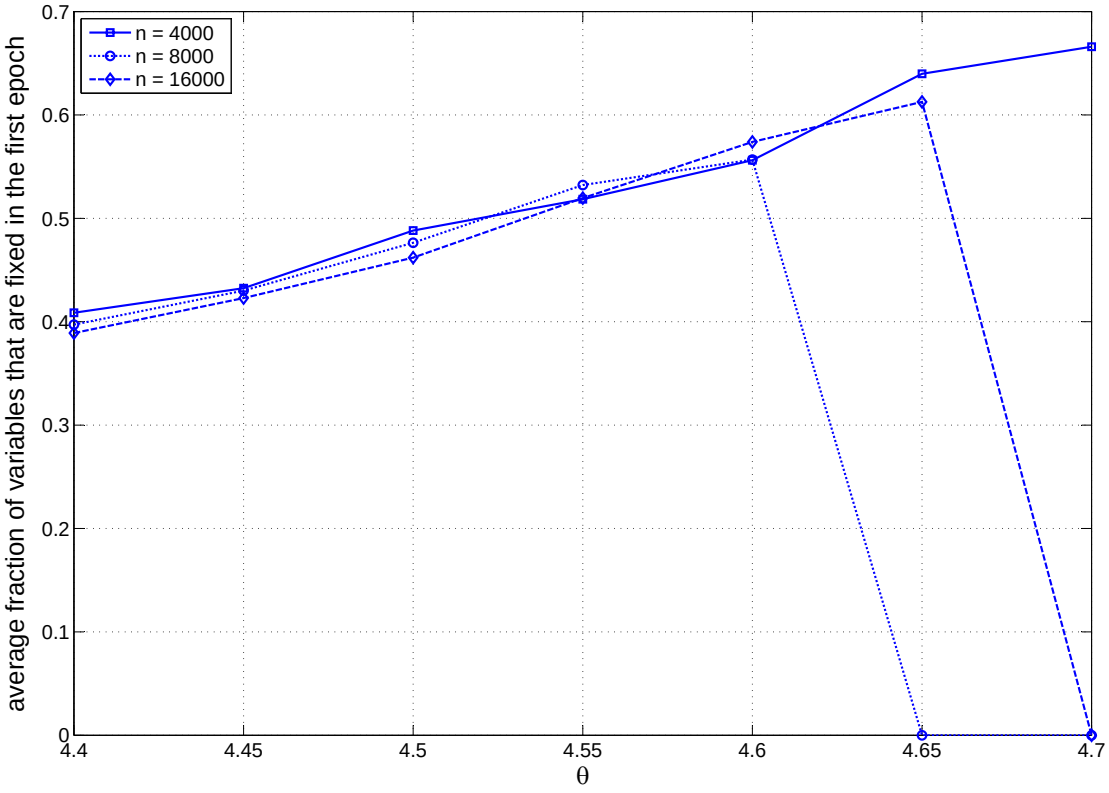


Figure 6.7: The behavior of proposed algorithm in the first epoch.

6.3.3 Performance of Proposed Algorithm

For each choice of graph ensemble, we ran our algorithm to color 50 randomly chosen graphs. The relative frequency at which the problem is solved is plotted in Figures 6.11, 6.12, and 6.13. In each figure, the performance of the BMPWZ algorithm is also plotted for comparison.

Across the three figures, it can be seen that as n increases from 4000 to 8000 and 16000, the proposed algorithm exhibits sharper threshold effect, similar to the BMPWZ algorithm. This is due to the fact that when the graph size increases, every graph drawn at random has statistically similar structure and the proposed algorithm performs similarly on them.

The most distinctive phenomenon in these figures is that the two performance curves in

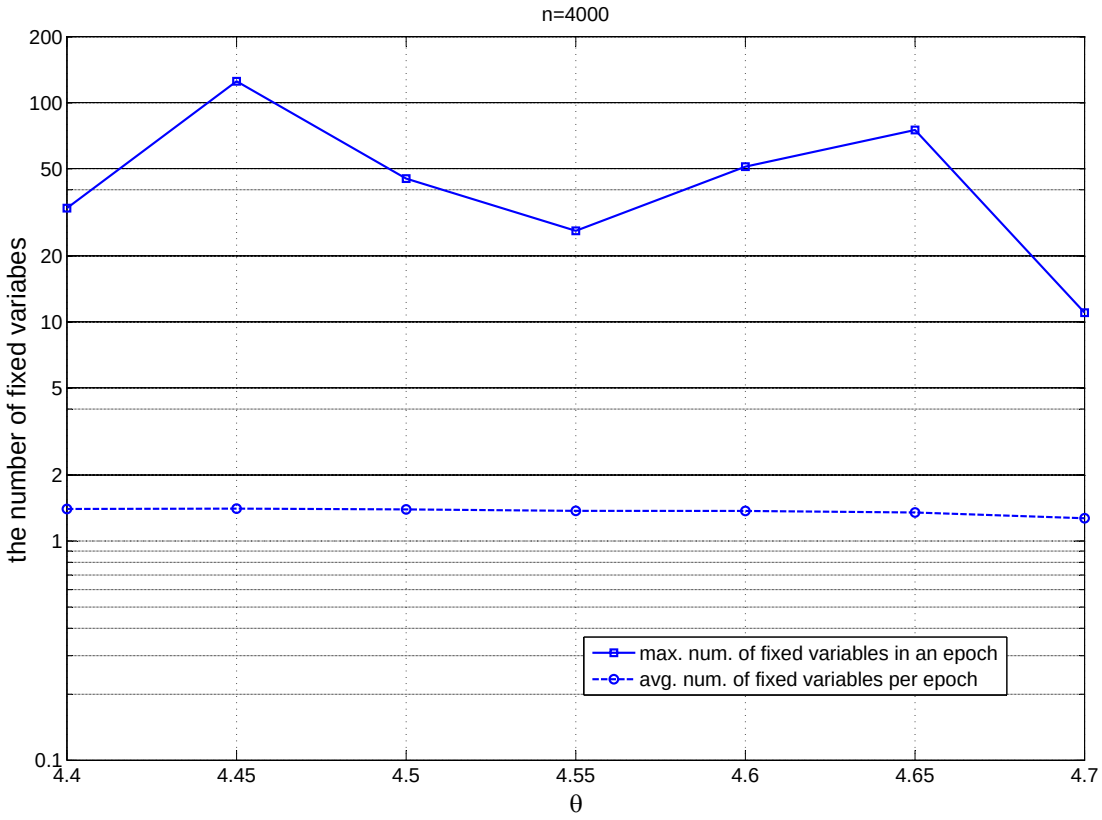


Figure 6.8: The behavior of proposed algorithm in later epoches for $n = 4000$.

each of the figures cross. That is, at higher values of the average vertex degree θ, the proposed algorithm performs superior to the BMPWZ algorithm, i.e., it is more capable of solving "harder" problems than the BMPWZ algorithm. However at low θ values, the proposed algorithm appears to perform inferior, in which case the frequency of solving a problem is bounded below 0.9 or so. We next briefly comment on this observation.

We note that the SP component of the BMPWZ algorithm may in fact also be viewed as weighted PTP with (α, β, γ) set to $(0, 0, 0)$. However, when we ran our algorithm using $(0, 0, 0)$ as the setting of (α, β, γ), our algorithm resulted in poorer performance than the BMPWZ algorithm (data not shown). This suggests that the decimation procedures (which also implicitly performs local search) appear to underperform the decimation and local search component of

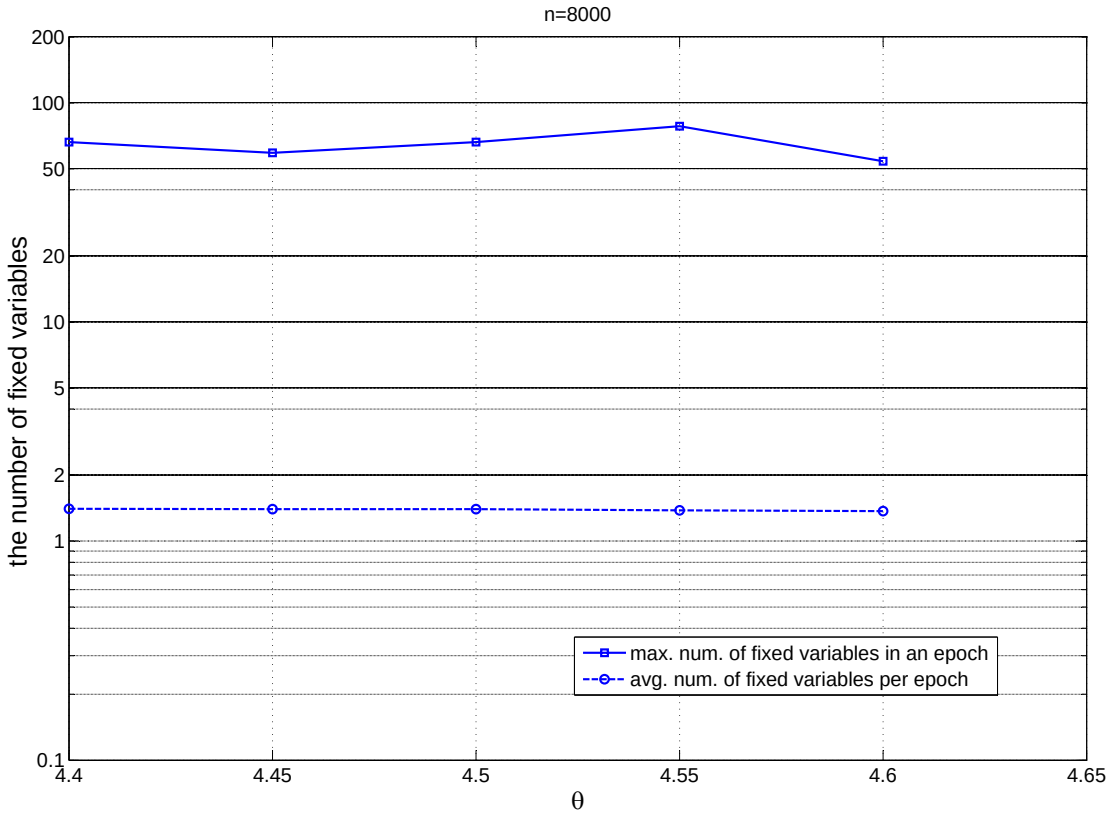


Figure 6.9: The behavior of proposed algorithm in later epoches for $n = 8000$.

the BMPWZ algorithm. This reasoning is consistent with and further justified by the observations described in the previous subsection. Indeed, in [10], the proposed decimation rule is quite sophisticated, and partly due to such sophistication, it is unfortunate that our best attempts to implement the decimation rules in the BMPWZ algorithm could not reproduce the reported results in [10].

Nevertheless, the performance comparison between the proposed algorithm and the BMPWZ algorithm does indicate that when the suboptimality of our decimation rules becomes less dominant, the proposed algorithm begins to outperform the BMPWZ algorithm. This should convincingly suggest that generalizing SP algorithms to weighted PTP allows potential performance improvements and that the solvability of random 3-COL problems using SP algorithms

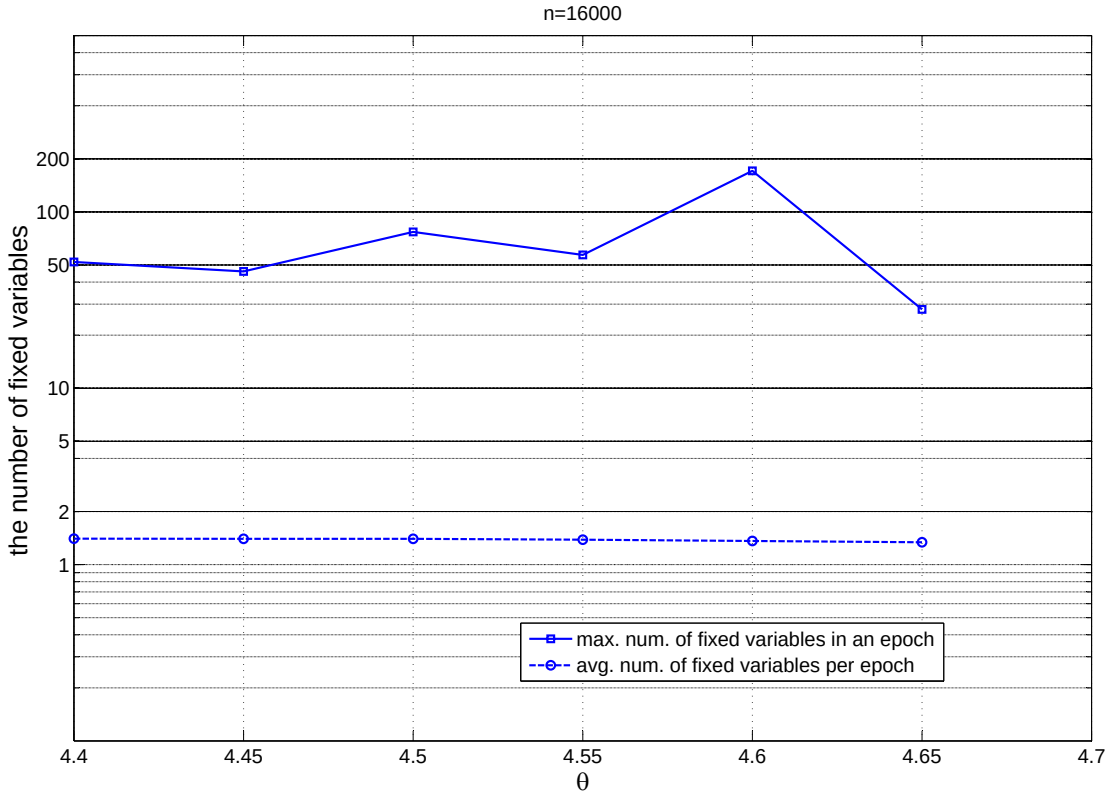


Figure 6.10: The behavior of proposed algorithm in later epoches for $n = 16000$.

may be pushed further by a careful design of weighted PTP based algorithms.

6.4 Concluding Remarks

In this chapter we have proposed a simple algorithm based on weighted PTP, or generalized SP, for 3-COL problems. With a heuristically selected parameter setting, the algorithm performs better on harder problems than the existing SP coloring algorithm, the BMPWZ algorithm of [10]. For easier problems, however, the BMPWZ algorithm still performs better, which is expected to result from less delicate implementation of decimation and local search in our algorithm. With further tuning of the parameters and better design of the decimation proce-

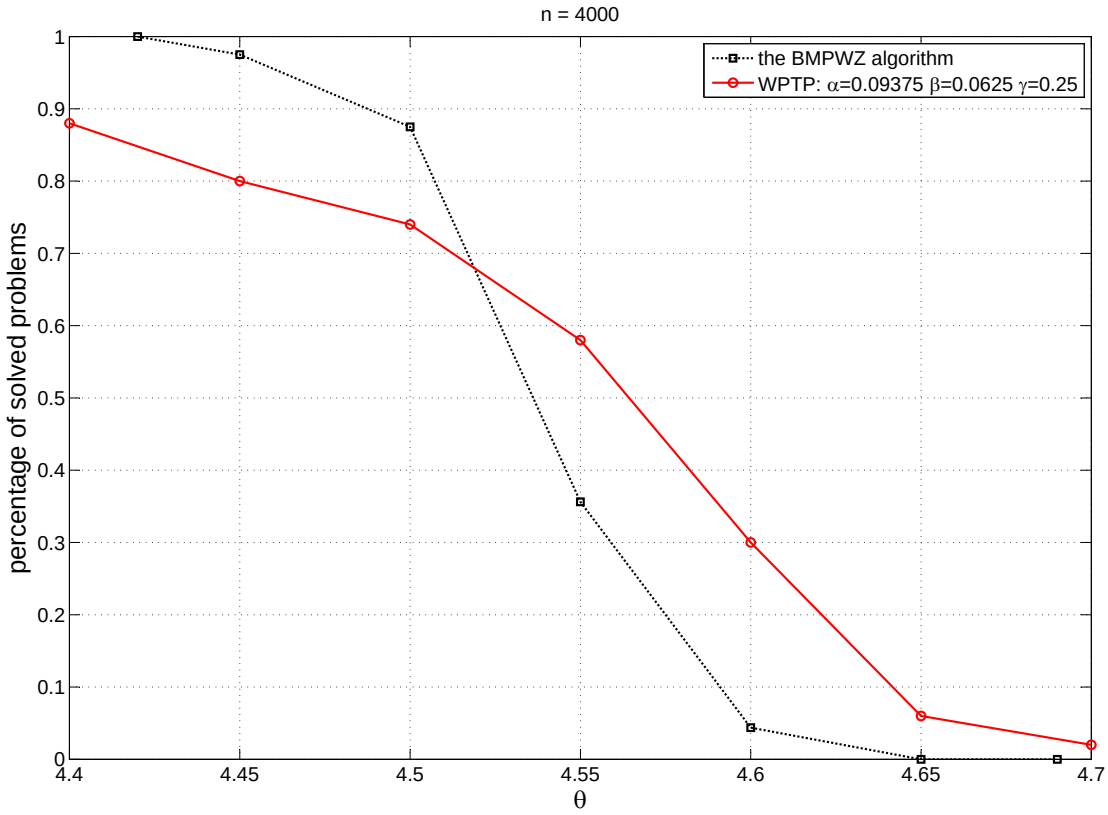


Figure 6.11: The relative frequencies at which an Erdös-Renyi random graph with $n = 4000$ is colored by the proposed algorithm and by the BMPWZ algorithm.

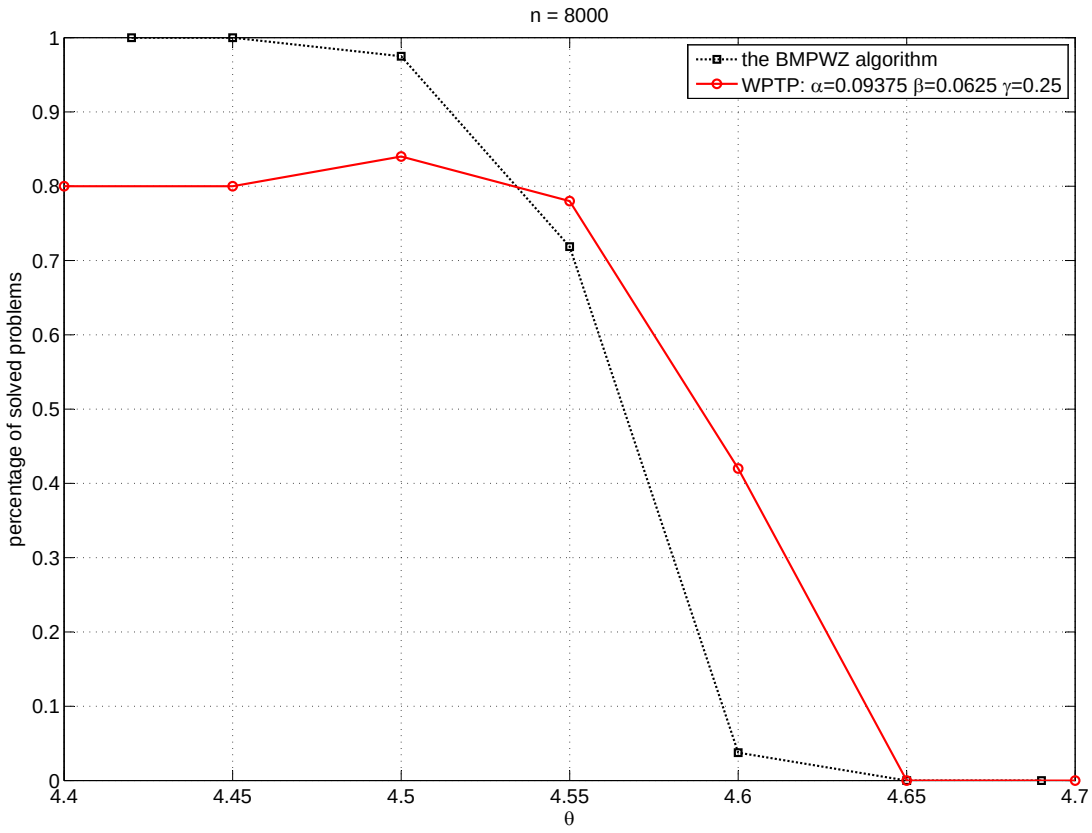


Figure 6.12: The relative frequencies at which an Erdös-Renyi random graph with $n = 8000$ is colored by the proposed algorithm and by the BMPWZ algorithm.

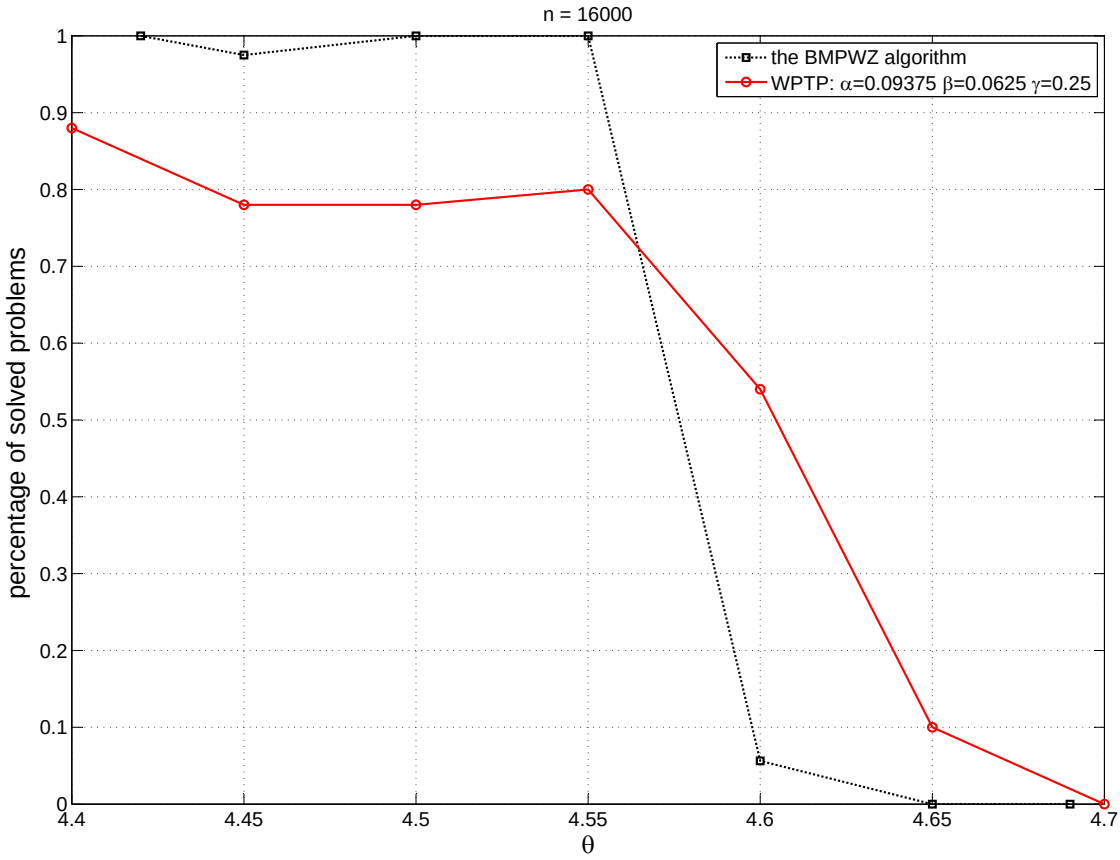


Figure 6.13: The relative frequencies at which an Erdös-Renyi random graph with $n = 16000$ is colored by the proposed algorithm and by the BMPWZ algorithm.

dure and local search components of the proposed algorithm, one can anticipate the proposed algorithm to outperform the BMPWZ algorithm consistently over a larger range of problem hardness. Nevertheless, the proposed algorithm has provided a convincing proof of concept for weighted PTP based CSP solvers and suggested the potential power of the generalized SP introduced in this thesis.

Chapter 7

Conclusions and Discussions

7.1 Thesis Summary

In this thesis, survey propagation (SP) is generalized to what we call the weighted probabilistic token passing (weighted PTP) algorithm. Weighted PTP has a simple and intuitive probabilistic interpretation and can be applied to arbitrary constraint satisfaction problems. Earlier SP algorithms presented in the literature for k-SAT problems and 3-COL problems are shown as special cases of the weighted PTP algorithms. Preliminary results on the dynamics of weighted PTP are also presented.

For a better understanding of SP algorithms, the question whether SP may in general be regarded as belief propagation (BP) is also investigated. We show that although when applied to k-SAT problems, SP and generalized SP (weighted PTP) may be derived from BP on a properly defined Markov Random Field, the claim that SP is BP is in general false. Additional conditions are required for the reduction of SP from BP. Such conditions are also established in this work.

Using a simple proof-of-concept implementation of a weighted PTP based graph-coloring algorithm, we show that generalized SP may potentially bring performance advantages over the existing SP algorithms.

7.2 Discussions

We would like to remark that PTP and weighted PTP formalisms in this thesis merely serve to give a more intuitive and easy explanation of SP algorithms and that we have no intention to use these names to replace the "SP" terminologies that have been established in the statistical physics community. In addition, to respect the history of SP algorithms and to include the concurrent developments in the field relating to this thesis, we list some important literature and their relation to this work.

- SP was first introduced in [29] for solving random k-SAT problems, which corresponds to PTP and weighted PTP in this work. Specifically, for the weighted version of the algorithm, a specific set of weights were used.

- Non-weighted version of SP (corresponding to PTP of this thesis) was introduced in [8], where the notion of token in this thesis was referred to as a "warning".

- DTP in this thesis was previously presented as "warning propagation" in [9].

- The first published discussion on the connection between SP and BP for k-SAT problems may be found in [11], which was later generalized in [23] asserting that weighted SP is BP for k-SAT problems.

- Reduction of SP from BP for general CSPs has also been studied recently in [25], where the authors use a similar MRF formalism over the fixed-point max-product messages on the factor graph representing the CSP. The authors of [25] show that under suitable conditions ([25] Equation (19.27)), carrying similar effects as the state-decoupling condition and the local compatibility condition of this thesis), standard (non-weighted) SP may be reduced from BP on their MRF formalism. In addition to discussing the connection between SP and BP, [25] provides a comprehensive treatment of cavity methods in statistical physics which lead to powerful analytic and algorithmic tools in information theory and computation.

It is worth noting that our answer to whether SP is BP is only restricted to the MRF formalism in the style of [23]. Although this restriction is not completely satisfactory, it appears to us that such an MRF formalism, particularly using Forney graphs, is the most natural in light of the natural correspondence between the states in the MRF and the SP messages (namely that left states correspond to the "intentions" of variables and right states correspond to the "commands" of the constraints). An additional and perhaps even stronger justification of this MRF is its combinatorial descriptive power as is elaborated in [23] for k-SAT problems, which — using the terminology of this thesis — captures the connectivity of the solution in the space of all "rectangles". In fact, we conjecture that further investigation of this perspective may provide useful insights into the algorithm design for solving hard instances of CSPs, whether or not SP or BP is considered as the choice of algorithms. [1]

Further we note that the BP algorithm has been understood as a special case of Generalized Belief Propagation (GBP) [41]. In that perspective, BP may be derived from iterative minimization of the Bethe-approximation of the notion of free energy [41]. The framework of GBP allows a variety of ways (unified under the notion of "region graphs") to approximate the free energy whereby leading to a much richer family of BP-like algorithms. Given the results of this work, one may not want to exclude the possibility that certain choice of free-energy approximation allows the corresponding GBP to reduce to SP algorithms for general CSPs. Research along that direction may still be of interest.

Finally, characterizing the dynamics of generalized SP and understanding the role of decimation in solving CSPs appear to be the next important problems in research on SP algorithms. Progresses along such directions are expected to help the design of SP algorithms and decimation rules and to significantly extend the applicability and effectiveness of SP solvers for CSP problems.

[1] In [23], under the MRF formalism, Gibbs sampling-based approach has also been presented as an algorithm for solving random k-SAT problems.

Bibliography

[1] D. Achlioptas and C. Moore. Random k-SAT: two moments suffice to cross a sharp threshold. *SIAM Journal on Computing*, 36:740–762, 2006.

[2] D. Achlioptas, A. Naor, and Y. Peres. Rigorous location of phase transitions in hard optimization problems. *Nature*, 435:759–764, 2005.

[3] D. Achlioptas and Y. Perez. The threshold for random k-SAT is $2^k log 2 - o(k)$. *Journal-American Mathematical Society*, 17:947–974, 2004.

[4] S. M. Aji and R. J. McEliece. The generalized distributive law. *IEEE Trans. Inform. Theory*, 46(2):325–343, Mar. 2000.

[5] F. Altarelli, R. Monasson, and F. Zamponi. Relationship between clustering and algorithmic phase transitions in the random k-XORSAT model and its NP-complete extensions. *Journal of Physics*, 40:867–886, 2007.

[6] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv. Optimal decoding of linear codes for minimizing symbol error rate. *IEEE Trans. Inform. Theory*, 20(2):284–287, March 1974.

[7] C. Berrou, A. Glavieux, and P. Thitimajshima. Near Shannon limit error-correcting coding and decoding: Turbo codes. *IEEE Trans. Comm.*, 44(10):1261–1271, Oct. 1996.

[8] A. Braunstein, M. Mézard, M. Weigt, and R. Zecchina. Constraint satisfaction by survey propagation. In *Computational Complexity and Statistical Physics*. Oxford University Press, 2003. Edited by C. Moore, G. Istrate and A. Percus.

[9] A. Braunstein, M. Mézard, and R. Zecchina. Survey propagation: an algorithm for satisfiability. *Random Structures and Algorithms*, 27:201–226, 2005.

[10] A. Braunstein, R. Mulet, A. Pagnani, M. Weigt, and R. Zecchina. Polynomial iterative algorithms for coloring and analyzing random graphs. *Physical Review E*, 68(3):036702, 2003.

[11] A. Braunstein and R. Zecchina. Survey propagation as local equilibrium equations. *J. Stat. Mech: Theory and Experiments (JSTAT)*, page 06007, June 2004.

[12] V. Chvátal and B. Reed. Mick gets some (the odds are on his side). In *the 33rd IEEE Symposium on Foundations of Computer Science (FOCS)*, pages 620–627, Pittsburgh, 1992.

[13] S. A. Cook. The complexity of theorem-proving procedures. In *3rd Annual ACM Symposium on Theory of Computing*, pages 151–158, 1971.

[14] O. Dubois and Y. Boufkhad. A general upper bound for the satisfiability threshold of random r-SAT formulae. *Journal of Algorithms*, 24:395–420, 1997.

[15] O. Dubois, Y. Boufkhad, and J. Mandler. Typical random 3-sat formulae and the satisfiability threshold. In *11'th SODA*, pages 126–127, 2000.

[16] G. D. Forney Jr. The Viterbi algorithm. *Proc. IEEE*, 61(3):268–278, March 1973.

[17] G. D. Forney Jr. Codes on graphs: normal realizations. *IEEE Trans. Inform. Theory*, 47(2):520–548, Feb. 2001.

[18] S. Franz, M. Leone, A. Montanari, and F. Ricci-Tersenghi. Dynamic phase transition for decoding algorithms. *Phys. Rev. E*, 22:046120, 2002.

[19] R. Kindermann and J. L. Snell. *Markov Random Fields and Their Applications*. American Mathematical Society, Providence, RI, 1980.

[20] L. M. Kirousis, E. Kranakis, D. Krizanc, and Y. C. Stamatiou. Approximating the unsatisfiability threshold of random formulas. *Random Structures and Algorithms*, 12:253–269, 1998.

[21] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger. Factor graphs and the sum-product algorithm. *IEEE Trans. Inform. Theory*, 47(2):498–519, Feb 2001.

[22] S. L. Lauritzen. *Graphical Models*. Oxford University Press, New York, 1st edition, 1996.

[23] E. Maneva, E. Mossel, and M. J. Wainwright. A new look at survey propagation and its generalizations. In *ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 1089–1098, 2005.

[24] R. J. McEliece, D. J. C. MacKay, and J. Cheng. Turbo decoding as an instance of Pearl's "belief propagation" algorithm. *IEEE JSAC*, 16(2):140–152, 1998.

[25] M. Mézard and A. Montanari. *Information, Physics, and Computation*. Oxford University Press, 2009.

[26] M. Mézard, T. Mora, and R. Zecchina. Clustering of solutions in the random satisfiability problem. *Phys. Rev. Lett.*, 94:197205, 2005.

[27] M. Mézard, M. Palassini, and O. Rivoire. Landscape of solutions in constraint satisfaction problems. *Phys. Rev. Lett.*, 95:200202, 2005.

[28] M. Mézard, G. Parisi, and R. Zecchina. Analytic and algorithmic solution of random satisfiability problems. *Science*, 5582:812–815, 2002.

[29] M. Mézard and R. Zecchina. Random K-satisfiability problem: from an analytic solution to an efficient algorithm. *Physical Review E*, 66:056126, 2002.

[30] R. Monasson and R. Zecchina. Tricritical points in random combinatorics: the $(2+p)$-SAT case. *Journal of Physics A*, 31:9209–0217, 1998.

[31] A. Montanari, F. Ricci-Tersenghi, and G. Semerjian. Clusters of solutions and replica symmetry breaking in random k-satisfiability. *Journal of Statistical Mechanics*, page 04004, 2008.

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

[33] T. Richardson and R. Urbanke. The capacity of low-density parity-check codes under message-passing decoding. *IEEE Trans. Inform. Theory*, 47(2):599–618, Feb. 2001.

[34] R. Tu, Y. Mao, and J. Zhao. On generalized survey propagation: normal realization and sum-product interpretation. In *Proc. IEEE Int. Symp. Inform. Theory*, pages 2042–2046, 2006.

[35] R. Tu, Y. Mao, and J. Zhao. Towards a unified solution for constraint-satisfaction problems: A survey-propagation approach based on normal realizations. In *Proc. 23rd Biennial Symp. Commun.*, pages 252–255, 2006.

[36] R. Tu, Y. Mao, and J. Zhao. Is SP BP? In *the 2007 IEEE Information Theory Workshop (ITW 2007)*, pages 236–241, 2007. Invited paper.

[37] R. Tu, Y. Mao, and J. Zhao. On the interpretation of survey propagation. In *Proc. 10th Canadian Workshop on Inform. Theory*, pages 77–80, 2007.

[38] R. Tu, Y. Mao, and J. Zhao. Survey propagation as probabilistic token passing. *IEICE Transactions on Information and Systems, Special Section on Foundations of Computer Science*, E91-D(2):231–233, February 2008. Letter.

[39] R. Tu, Y. Mao, and J. Zhao. Is SP BP? *IEEE Transactions on Information Theory*, (6):2999–3032, June 2010.

[40] M. J. Wainwright and E. Maneva. Lossy source encoding via message-passing and decimation over generalized codewords of LDGM codes. In *Proc. IEEE Int. Symp. Inform. Theory*, pages 1493–1497, Adelaide, Australia, 2005.

[41] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Trans. Inform. Theory*, 51(7):2282–2312, July 2005.

[42] W. Yu and M. Aleksic. Coding for the Blackwell channel: a survey propagation approach. In *Proc. IEEE Int. Symp. Inform. Theory*, pages 1583–1587, Adelaide, Australia, 2005.

[43] L. Zdeborová and F. Krzakala. Phase transitions in the coloring of random graphs. *Physical Review E*, 76:031131, 2007.