Message Passing and Combinatorial Optimization

by

Mohsen Ravanbakhsh

A thesis submitted in partial fulfillment of the requirements for the degree of

Doctor of Philosophy

Department of Computing Science

University of Alberta

Abstract

Graphical models use the intuitive and well-studied methods of graph theory to implicitly represent dependencies between variables in large systems. They can model the global behaviour of a complex system by specifying only local factors.This thesis studies inference in discrete graphical models from an "algebraic perspective" and the ways inference can be used to express and approximate $\mathbb{NP}$-hard combinatorial problems.

We investigate the complexity and reducibility of various inference problems, in part by organizing them in an inference hierarchy. We then investigate tractable approximations for a subset of these problems using distributive law in the form of message passing. The quality of the resulting message passing procedure, called Belief Propagation (BP), depends on the influence of loops in the graphical model. We contribute to three classes of approximations that improve BP for loopy graphs (I) loop correction techniques; (II) survey propagation, another message passing technique that surpasses BP in some settings; and (III) hybrid methods that interpolate between deterministic message passing and Markov Chain Monte Carlo inference.

We then review the existing message passing solutions and provide novel graphical models and inference techniques for combinatorial problems under three broad classes: (I) constraint satisfaction problems (CSPs) such as satisfiability, coloring, packing, set / clique-cover and dominating / independent set and their optimization counterparts; (II) clustering problems such as hierarchical clustering, K-median, K-clustering, K-center and modularity optimization; (III) problems over permutations including (bottleneck) assignment, graph "morphisms" and alignment, finding symmetries and (bottleneck) traveling salesman problem. In many cases we show that message passing is able to find solutions that are either near optimal or favourably compare with today's state-of-the-art approaches.

To the memory of my grandparents.

Acknowledgements

Many have helped me survive and grow during my graduate studies. First, I would like to thank my supervisor Russell Greiner, for all I have learned from him, most of all for teaching me the value of common sense in academia and also for granting me a rare freedom in research. During these years, I had the chance to learn from many great minds in Alberta and elsewhere. It has been a pleasure working with David Wishart, Brendan Frey, Jack Tuszynski and Barnabás Póczos. I am also grateful to my committee members, Dale Schuurmans, Csaba Szepesvári, Mohammad Salavatipour and my external examiner Cristopher Moore for their valuable feedback.

I have enjoyed many friendly conversations with colleagues: Babak Alipanahi, Nasimeh Asgarian, Trent Bjorndahl, Kit Chen, Andrew Delong, Jason Grant, Bret Hoehn, Sheehan Khan, Philip Liu, Alireza Makhzani, Rupasri Mandal, James Neufeld, Christopher Srinivasa, Mike Wilson, Chunnam Yu and others.

I am thankful to my friends: Amir, Amir-Mashoud, Amir-Ali, Amin, Arash, Arezoo, Azad, Babak(s), Fariba, Farzaneh, Hootan, Hossein, Kent, Kiana, Maria, Mariana, Meys, Meysam, Mohsen, Mohammad, Neda, Niousha, Pirooz, Sadaf, Saeed, Saman, Shaham, Sharron, Stuart, Yasin, Yavar and other for all the good time in Canada. Finally I like to thank my family and specially Reihaneh and Tare for their kindness and constant support.

Here, I acknowledge the help and mental stimulation from the playful minds on the Internet, from Reddit to stackexchange and the funding and computational/technical support from computing science help desk, Alberta Innovates Technology Futures, Alberta Innovates Center for Machine Learning and Compute Canada.

Contents

List of tables

List of figures

Notation

x, y, z **(lower case roman letters)** single variables.

$\mathcal{X}$, $\mathcal{Y}$,$\mathcal{C}$ **(upper case caligraphic letters)** sets.

I, J, K **(upper case roman letters)** sets of variable indices.

f, q, p **(lower case sans serif letters)** functions over discrete domains (equivalent to arrays/tensors).

P, Q, X **(lower case sans serif letters)** functionals.

$\underline{x}, \underline{\mathsf{q}}, \underline{\mathsf{P}}$ **(underline)** "tuples" of variables, functions or functionals.

$\mathbf{A}, \mathbf{B}$ **(bold upper case roman letters)** Matrices.

$\mathsf{q}_{i \to \mathrm{I}}, \underline{\widetilde{\mathsf{P}}}_{\partial i \to \mathrm{I}}, \mathsf{S}_{\mathrm{I} \to i}$ **(arrow in the subscript)** (tuple of) "messages" as functions or functionals.

$\mathsf{p}, \widehat{\mathsf{p}}_{\mathrm{I} \to i}, \mathsf{P}_{I \to i}$ **(variants of letter p)** "normalized" marginals or messages as functions or functionals.

$\mathbb{P}, \mathbb{NP}, \mathbb{PP}, \#\mathbb{P}, \mathbb{PSPACE}$ **(blackboard bold letters)** complexity classes.

Σ, Φ, Ψ **(capital greek letters)** inference families.

Introduction

Many complicated systems can be modeled as a graphical structure with interacting local functions. Many fields have (almost independently) discovered this: graphical models have been used in bioinformatics (protein folding, medical imaging and spectroscopy, pedagogy trees, regulatory networks [27, 178, 224, 255, 323]), neuroscience (formation of associative memory and neuroplasticity [10, 65]), communication theory (low density parity check codes [106, 290]), statistical physics (physics of dense matter and spin-glass theory [211]), image processing (inpainting, stereo/texture reconstruction, denoising and super-resolution [98, 101]), compressed sensing [84], robotics [294] (particle filters), sensor networks [76, 143], social networks [203, 307], natural language processing [200], speech recognition [73, 131], artificial intelligence (artificial neural networks, Bayesian networks [244, 316]) and **combinatorial optimization**. This thesis is concerned with the application of graphical models in solving combinatorial optimization problems [122, 222, 287], which broadly put, seeks *an "optimal" assignment to a discrete set of variables, where a brute force approach is infeasible.*

To see how the decomposition offered by a graphical model can model a complex system, consider a joint distribution over 200 binary variables. A naive way to represent this would require a table with 2^{200} entries. However if variables are conditionally independent such that their dependence structure forms a tree, we can exactly represent the joint distribution using only 200×2^2 values. Operations such as marginalization, which require computation time linear in the original size, are now reduced to local computation in the form of message passing on this structure (*i.e.*, tree), which in this case, reduces the cost to linear in the new exponentially smaller size. It turns out even if the dependency structure has loops, we can use message passing to perform "approximate" inference.

Moreover, we approach the problem of inference from an algebraic point of view [7]. This is in contrast to the variational perspective on local computation [303]. These two perspectives are to some extent "residuals" from the different origins of research in AI and statistical physics.

In the statistical study of physical systems, the Boltzmann distribution relates the probability of each state of a physical system to its energy, which is often decomposed due to local interactions [208, 211]. These studies have been often interested in modeling systems at the thermodynamic

limit of infinite variables and the average behaviour through the study of random ensembles. Inference techniques with this origin (*e.g.*, mean-field and cavity methods) are often asymptotically exact under these assumptions. Most importantly these studies have reduced inference to optimization through the notion of free energy –a.k.a. variational approach.

In contrast, graphical models in the AI community have emerged in the study of knowledge representation and reasoning under uncertainty [245]. These advances are characterized by their attention to the theory of computation and logic [17], where interest in computational (as opposed to analytical) solutions has motivated the study of approximability, computational complexity [74, 267] and invention of inference techniques such as belief propagation that are efficient and exact on tree structures. Also, these studies have lead to algebraic abstractions in modeling systems that allow local computation [185, 279].

The common foundation underlying these two approaches is information theory, where derivation of probabilistic principles from logical axioms [146] leads to notions such as entropy and divergences that are closely linked to their physical counter-parts *i.e.*, entropy and free energies in physical systems. At a less abstract level, it was shown that inference techniques in AI and communication are attempting to minimize (approximations to) free energy [6, 326].

Another exchange of ideas between the two fields was in the study of critical phenomenon in random constraint satisfaction problems by both computer scientists and physicists [103, 215, 216]; satisfiability is at the heart of theory of computation and an important topic to investigate reasoning in AI. On the other hand, the study of critical phenomena and phase transitions is central in statistical physics of disordered systems. This was culminated when a variational analysis lead to discovery of survey propagation [213] for constraint satisfaction, which significantly advanced the state-of-the-art in solving random satisfiability problems.

Despite this convergence, variational and algebraic perspectives are to some extent complementary – *e.g.*, the variational approach does not extend beyond (log) probabilities, while the algebraic approach cannot justify application of message passing to graphs with loops. Although we briefly review the variational perspective, this thesis is mostly concerned with the algebraic perspective. In particular, rather than the study of phase transitions and the behaviour of the set of solutions for combinatorial problems, we are concerned with finding solutions to individual instances.

Part I starts by expressing the general form of inference, proposes a novel inference hierarchy and studies its complexity in chapter 1. Here, we also show how some of these problems are reducible to others and introduce the algebraic structures that make efficient inference possible. The general form of notation and the reductions that are proposed in this chapter are used in later chapters.

Chapter 2 studies some forms of approximate inference, by first introducing belief propagation. It then considers the problems with intractably large number of factors and factors with large

cardinality, then proposes/reviews solutions to both problems. We then study different modes of inference as optimization and review alternatives such as convergent procedures and convex and linear programming relaxations for some inference classes in the inference hierarchy. Standard message passing using belief propagation is only guaranteed to be exact if the graphical structure has no loops. This optimization perspective (a.k.a. variational perspective) has also led to design of approximate inference techniques that account for short loops in the graph. A different family of loop correction techniques can account for long loops by taking message dependencies into account. This chapter reviews these methods and introduces a novel loop correction scheme that can account for both short and long loops, resulting in more accurate inference over difficult instances.

Message passing over loopy graphs can be seen as a fixed point iteration procedure, and the existence of loops means there may be more than one fixed point. Therefore an alternative to loop correction is to in some way incorporate all fixed points. This can be performed also by a message passing procedure, known as survey propagation. The next section of this chapter introduces survey propagation from a novel algebraic perspective that enables performing inference on the set of fixed points. Another major approach to inference is offered by Markov Chain Monte Carlo (MCMC) techniques. After a minimal review of MCMC, the final section of this chapter introduces a hybrid inference procedure, called perturbed belief propagation that interpolates between belief propagation and Gibbs sampling. We show that this technique can outperform both belief propagation and Gibbs sampling in particular settings.

Part II of this thesis uses the inference techniques derived in the first part to solve a wide range of combinatorial problems. We review the existing message passing solutions and provide novel formulations for three broad classes of problems: 1) constraint satisfaction problems (CSPs), 2) clustering problems and 3) combinatorial problems over permutations.

In particular, in chapter 3 we use perturbed belief propagation and perturbed survey propagation to obtain state-of-the-art performance in random satisfiability and coloring problems. We also introduce novel message passing solutions and review the existing methods for sphere packing, set-cover, clique-cover, dominating-set and independent-set and several of their optimization counterparts. By applying perturbed belief propagation to graphical representation of packing problem, we are able to compute long "optimal" nonlinear binary codes with large number of digits.

Chapter 4 proposes message passing solutions to several clustering problems such as K-clustering, K-center and Modularity optimization and shows that message passing is able to find near-optimal solutions on moderate instances of these problems. Here, we also review the previous approaches to K-median and hierarchical clustering and also the related graphical models for minimum spanning tree and prize-collecting Steiner tree.

Chapter 5 deals with combinatorial problems over permutations, by first reviewing the existing

graphical models for matching, approximation of permanent, and graph alignment and introducing two novel message passing solutions for min-sum and min-max versions of traveling salesman problem (a.k.a. bottleneck TSP). We then study graph matching problems, including (sub-)graph isomorphism, monomorphism, homomorphism, graph alignment and "approximate" symmetries. In particular, in the study of graph homomorphism we show that its graphical model generalizes that of of several other problems, including Hamiltonian cycle, clique problem and coloring. We further show how graph homomorphism can be used as a surrogate for isomorphism to find symmetries.

Contributions and acknowledgment

The results in this thesis are a joint work with my supervisor Dr. Greiner and other researchers. In detail, the algebraic approach to inference is presented in [256]; the loop correction ideas are published in [254]; perturbation schemes for CSP are presented in [253]; performing min-max inference was first suggested by Dr. Brendan Frey and Christopher Srinivasa, and many of the related ideas including min-max reductions are presented in our joint paper [258]. Finally, the augmentation scheme for TSP and Modularity maximization is discussed in [257].

The contribution of this thesis, including all the published work is as follows:

- Generalization of inference problems in graphical models including:
  - The inference hierarchy.
  - The limit of distributive law on tree structures.
  - All the theorems, propositions and claims on complexity of inference, including
    - $\mathbb{NP}$-hardness of inference in general commutative semirings.
- A unified treatment of different modes of inference over factor-graphs and identification of their key properties (*e.g.*, significance of inverse operator) in several settings including:
  - Loop correction schemes.
  - Survey propagation equations.
- Reduction of min-max inference to min-sum and sum-product inference.
- Simplified form of loop correction in Markov networks and their generalization to incorporate short loops over regions.
- A novel algebraic perspective on survey propagation.
- Perturbed BP and perturbed SP and their application to constraint satisfaction problems.
- Factor-graph augmentation for inference over intractably large number of constraints.
- Factor-graph formulation for several combinatorial problems including
  - Clique-cover.
  - Independent-set, set-cover and vertex cover.
  - Dominating-set and packing (the binary-variable model)

- Packing with hamming distances
- K-center problem, K-clustering and clique model for modularity optimization.
- TSP and bottleneck TSP.

– The general framework for study of graph matching, including

- Subgraph isomorphism.[1]
- Study of message passing for Homomorphism and finding approximate symmetries.
- Graph alignment with a diverse set of penalties.

[1] Although some previous work [44] claim to address the same problem, we note that their formulation is for subgraph monomorphism rather than isomorphism.

Part I

Inference by message passing

This part of the thesis first studies the representation formalism, hierarchy of inference problems, reducibilities and the underlying algebraic structure that allows efficient inference in the form of message passing in graphical models, in chapter 1. By viewing inference under different lights, we then review/introduce procedures that allow better approximations in chapter 2.

Chapter 1

Representation, complexity and reducibility

In this chapter, we use a simple algebraic structure – *i.e.*, commutative semigroup – to express a general form for inference in graphical models. To this end, we first introduce the factor-graph representation and formalize inference in section 1.1. Section 1.2 focuses on four operations defined by summation, multiplication, minimization and maximization, to construct a hierarchy of inference problems within $\mathbb{PSPACE}$, such that the problems in the same class of the hierarchy belong to the same complexity class. Here, we encounter some new inference problems and establish the completeness of problems at lower levels of hierarchy w.r.t. their complexity classes. In section 1.3 we augment our simple structures with two properties to obtain message passing on commutative semirings. Here, we also observe that replacing a semigroup with an Abelian group, gives us normalized marginalization as a form of inference inquiry. Here, we show that inference in any commutative semiring is $\mathbb{NP}$-hard and postpone further investigation of message passing to the next chapter. Section 1.4 shows how some of the inference problems introduced so far are reducible to others.

1.1 The problem of inference

We use commutative semigroups to both define what a graphical model represents and also to define inference over this graphical model. The idea of using structures such as semigroups, monoids and semirings in expressing inference has a long history[33, 185, 275]. Our approach, based on factor-graphs [181] and commutative semigroups, generalizes a variety of previous frameworks, including Markov networks [68], Bayesian networks [243], Forney graphs [100], hybrid models [83], influence diagrams [137] and valuation networks [278].

In particular, the combination of factor-graphs and semigroups that we consider here gener-

alizes the plausibility, feasibility and utility framework of Pralet et al. [250], which is explicitly reduced to the graphical models mentioned above and many more. The main difference in our approach is in keeping the framework free of semantics (*e.g.*, decision and chance variables, utilities, constraints), that are often associated with variables, factors and operations, without changing the expressive power. These notions can later be associated with individual inference problems to help with interpretation.

Definition 1.1.1. A **commutative semigroup** is a pair $\mathscr{G} = (\mathcal{Y}^*, \otimes)$, where $\mathcal{Y}^*$ is a set and $\otimes : \mathcal{Y}^* \times \mathcal{Y}^* \to \mathcal{Y}^*$ is a binary operation that is (I) associative: $a \otimes (b \otimes c) = (a \otimes b) \otimes c$ and (II) commutative: $a \otimes b = b \otimes a$ for all $a, b, c \in \mathcal{Y}^*$. A **commutative monoid** is a commutative semigroup plus an identity element $\overset{\otimes}{1}$ such that $a \otimes \overset{\otimes}{1} = a$. If every element $a \in \mathcal{Y}^*$ has an inverse a^{-1} (often written $\frac{1}{a}$), such that $a \otimes a^{-1} = \overset{\otimes}{1}$, and $a \otimes \overset{\otimes}{1} = a$, the commutative monoid is an **Abelian group**.

Here, the associativity and commutativity properties of a commutative semigroup make the operations invariant to the order of elements. In general, these properties are not "vital" and one may define inference starting from a *magma*.[1]

Example 1.1.1. Some examples of semigroups are:

- The set of strings with the concatenation operation forms a semigroup with the empty string as the identity element. However this semigroup is not commutative.
- The set of natural numbers $\mathcal{N}$ with summation defines a commutative semigroup.
- Integers modulo n with addition defines an Abelian group.
- The power-set $2^{\mathcal{S}}$ of any set $\mathcal{S}$, with intersection operation defines a commutative semigroup with $\mathcal{S}$ as its identity element.
- The set of natural numbers with greatest common divisor defines a commutative monoid with 0 as its identity. In fact any semilattice is a commutative semigroup [79].
- Given two commutative semigroups on two sets $\mathcal{Y}^*$ and $\mathcal{Z}^*$, their Cartesian product is also a commutative semigroup.

Let $\underline{x} = (x_1, \ldots, x_N)$ be a tuple of N discrete variables $x_i \in \mathcal{X}_i$, where $\mathcal{X}_i$ is the domain of x_i and $\underline{x} \in \mathcal{X} = \mathcal{X}_1 \times \ldots \times \mathcal{X}_N$. Let $\mathrm{I} \subseteq \mathcal{N} = \{1, 2, \ldots, N\}$ denote a subset of variable indices and $\underline{x}_\mathrm{I} = \{x_i \mid i \in \mathrm{I}\} \in \mathcal{X}_\mathrm{I}$ be the tuple of variables in $\underline{x}$ indexed by the subset I. A factor $\mathrm{f}_\mathrm{I} : \mathcal{X}_\mathrm{I} \to \mathcal{Y}_\mathrm{I}$ is a function over a subset of variables and $\mathcal{Y}_\mathrm{I} = \{\mathrm{f}_\mathrm{I}(\underline{x}_\mathrm{I}) \mid \underline{x}_\mathrm{I} \in \mathcal{X}_\mathrm{I}\}$ is the range of this factor.

Definition 1.1.2. A **factor-graph** is a pair $(\mathcal{F}, \mathscr{G})$ such that

[1] A magma [247] generalizes a semigroup, as it does not require associativity property nor an identity element. Inference in graphical models can be also extended to use magma (in definition 1.1.2). For this, the elements of $\mathcal{Y}^*$ and/or $\mathcal{X}$ should be ordered and/or parenthesized so as to avoid ambiguity in the order of pairwise operations over the set. Here, to avoid unnecessary complications, we confine our treatment to commutative semigroups.

- $\mathcal{F} = \{\mathrm{f_I}\}$ is a collection of factors with collective range $\mathcal{Y} = \bigcup_{\mathrm{I}} \mathcal{Y}_{\mathrm{I}}$.
- $|\mathcal{F}| = \mathrm{Poly}(N)$.
- $\mathrm{f_I}$ has a polynomial representation in N and it is possible to evaluate $\mathrm{f_I}(\underline{x}_{\mathrm{I}})$ $\forall \mathrm{I}, \underline{x}_{\mathrm{I}}$ in polynomial time.
- $\mathcal{G} = (\mathcal{Y}^*, \otimes)$ is a commutative semigroup, where $\mathcal{Y}^*$ is the closure of $\mathcal{Y}$ w.r.t. $\otimes$.

The factor-graph compactly represents the expanded (joint) form

$$\mathrm{q}(\underline{x}) = \bigotimes_{\mathrm{I}} \mathrm{f_I}(\underline{x}_{\mathrm{I}}) \tag{1.1}$$

Note that the connection between the set of factors $\mathcal{F}$ and the commutative semigroup is through the "range" of factors. The conditions of this definition are necessary and sufficient to 1) compactly represent a factor-graph and 2) evaluate the expanded form, $\mathrm{q}(\underline{x})$, in polynomial time. A stronger condition to ensure that a factor has a compact representation is $|\mathcal{X}_{\mathrm{I}}| = \mathrm{Poly}(N)$, which means $\mathrm{f_I}(\underline{x}_{\mathrm{I}})$ can be explicitly expressed for each $\underline{x}_{\mathrm{I}} \in \mathcal{X}_{\mathrm{I}}$ as an $|\mathrm{I}|$-dimensional array.

$\mathcal{F}$ can be conveniently represented as a bipartite graph that includes two sets of nodes: variable nodes x_i, and factor nodes I. A variable node i (note that we will often identify a variable x_i with its index "i") is connected to a factor node I if and only if $i \in \mathrm{I}$ –*i.e.*, I is a set that is also an index. We will use ∂ to denote the neighbours of a variable or factor node in the factor graph – that is $\partial \mathrm{I} = \{i \mid i \in \mathrm{I}\}$ (which is the set I) and $\partial i = \{\mathrm{I} \mid i \in \mathrm{I}\}$. Also, we use Δi to denote the **Markov blanket** of node x_i – *i.e.*, $\Delta i = \{j \in \partial \mathrm{I} \mid \mathrm{I} \in \partial i,\ j \neq i\}$.

Example 1.1.2. Figure 1.1 shows a factor-graph with 12 variables and 12 factors. Here $\underline{x} = (x_i, x_j, x_k, x_e, x_m, x_o, x_r, x_s, x_t, x_u, x_v, x_w)$, $\mathrm{I} = \partial \mathrm{I} = \{i, j, k\}$, $\underline{x}_{\mathrm{K}} = \underline{x}_{\{k,w,v\}}$ and $\partial j = \{\mathrm{I}, \mathrm{V}, \mathrm{W}\}$. Assuming $\mathcal{G}_e = (\mathbb{R}, \min)$, the expanded form represents

$$\mathrm{q}(\underline{x}) = \min\{\mathrm{f_I}(\underline{x}_{\mathrm{I}}), \mathrm{f_J}(\underline{x}_{\mathrm{J}}), \ldots, \mathrm{f_Z}(\underline{x}_{\mathrm{Z}})\}.$$

Now, assume that all variables are binary – *i.e.*, $\mathcal{X} = \{0,1\}^{12}$ and $\mathrm{q}(\underline{x})$ is 12-dimensional hypercube, with one assignment at each corner. Also assume all the factors count the number of non-zero variables – *e.g.*, for $\underline{z}_{\mathrm{W}} = (1,0,1) \in \mathcal{X}_{\mathrm{W}}$ we have $\mathrm{f_W}(\underline{z}_{\mathrm{W}}) = 2$. Then, for the complete assignment $\underline{z} = (0,1,0,1,0,1,0,1,0,1,0,1) \in \mathcal{X}$, it is easy to check that the expanded form is $\mathrm{q}(\underline{z}) = \min\{2,0,1,\ldots,1\} = 0$.

A marginalization operation shrinks the expanded form $\mathrm{q}(\underline{x})$ using another commutative semigroup with binary operation $\oplus$. Inference is a combination of an expansion and one or more marginalization operations, which can be computationally intractable due to the exponential size of the expanded form.

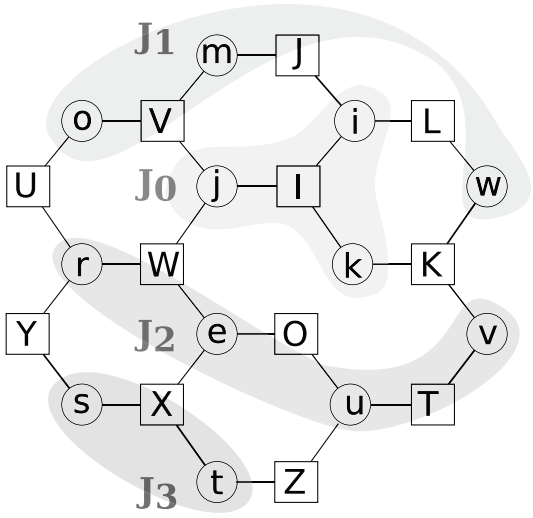


Figure 1.1: *A factor-graph with variables as circles and factors as squares.*

Definition 1.1.3. Given a function $\mathrm{q} : \mathcal{X}_\mathrm{J} \to \mathcal{Y}$, and a commutative semigroup $\mathscr{G} = (\mathcal{Y}^*, \oplus)$, where $\mathcal{Y}^*$ is the closure of $\mathcal{Y}$ w.r.t. $\oplus$, the marginal of q for $\mathrm{I} \subset \mathrm{J}$ is

$$\mathrm{q}(\underline{x}_{\mathrm{J}\setminus\mathrm{I}}) \stackrel{\text{def}}{=} \bigoplus_{\underline{x}_\mathrm{I}} \mathrm{q}(\underline{x}_\mathrm{J}) \tag{1.2}$$

where $\bigoplus_{\underline{x}_\mathrm{I}} \mathrm{q}(\underline{x}_\mathrm{J})$ is short for $\bigoplus_{\underline{x}_\mathrm{I} \in \mathcal{X}_\mathrm{I}} \mathrm{q}(\underline{x}_{\mathrm{J}\setminus\mathrm{I}}, \underline{x}_\mathrm{I})$, and it means to compute $\mathrm{q}(\underline{x}_{\mathrm{J}\setminus\mathrm{I}})$ for each $\underline{x}_{\mathrm{J}\setminus\mathrm{I}}$, one should perform the operation $\oplus$ over the set of all the assignments to the tuple $\underline{x}_\mathrm{I} \in \mathcal{X}_\mathrm{I}$.

We can think of $\mathrm{q}(\underline{x}_\mathrm{J})$ as a $|\mathrm{J}|$-dimensional tensor and marginalization as performing $\oplus$ operation over the axes in the set I. The result is another $|\mathrm{J} \setminus \mathrm{I}|$-dimensional tensor (or function) that we call the **marginal**. Here if the marginalization is over all the dimensions in J, we denote the marginal by $\mathrm{q}(\emptyset)$ instead of $\mathrm{q}(\underline{x}_\emptyset)$ and call it the **integral** of q.

Now we define an inference problem as a sequence of marginalizations over the expanded form of a factor-graph.

Definition 1.1.4. An **inference problem** seeks

$$\mathrm{q}(\underline{x}_{\mathrm{J}_0}) = \bigoplus_{\underline{x}_{\mathrm{J}_M}}^{M} \bigoplus_{\underline{x}_{\mathrm{J}_{M-1}}}^{M-1} \cdots \bigoplus_{\underline{x}_{\mathrm{J}_1}}^{1} \bigotimes_{\mathrm{I}} \mathrm{f}_\mathrm{I}(\underline{x}_\mathrm{I}) \tag{1.3}$$

where

- $\mathcal{Y}^*$ is the closure of $\mathcal{Y}$ (the collective range of factors), w.r.t. $\overset{1}{\oplus}, \ldots, \overset{M}{\oplus}$ and $\otimes$.
- $\mathscr{G}_m = (\mathcal{Y}^*, \overset{m}{\oplus})$ $\forall 1 \leq m \leq M$ and $\mathscr{G}_e = (\mathcal{Y}^*, \otimes)$ are all commutative semigroups.

- $\mathrm{J}_0, \ldots, \mathrm{J}_L$ partition the set of variable indices $\mathcal{N} = \{1, \ldots, N\}$.
- $\mathrm{q}(\underline{x}_{\mathrm{J}_0})$ has a polynomial representation in N – *i.e.*, $|\mathcal{X}_{\mathrm{J}_0}| = \mathrm{Poly}(N)$

Note that $\overset{1}{\oplus}, \ldots, \overset{M}{\oplus}$ refer to potentially different operations as each belongs to a different semigroup. When $\mathrm{J}_0 = \emptyset$, we call the inference problem **integration** (denoting the inquiry by $\mathrm{q}(\emptyset)$) and otherwise we call it **marginalization**. Here, having a constant sized J_0 is not always enough to ensure that $\mathrm{q}(\underline{x}_{\mathrm{J}_0})$ has a polynomial representation in N. This is because the size of $\mathrm{q}(\underline{x}_{\mathrm{J}_0})$ for any individual $\underline{x}_{\mathrm{J}_0} \in \mathcal{X}_{\mathrm{J}_0}$ may grow exponentially with N (*e.g.*, see claim 1.2.1). In the following we call $\mathcal{G}_e = (\mathcal{Y}^*, \otimes)$ the expansion semigroup and $\mathcal{G}_m = (\mathcal{Y}^*, \overset{m}{\oplus})$ $\forall 1 \leq m \leq M$ the marginalization semigroup.

Example 1.1.3. Going back to example 1.1.3, the shaded region in figure 1.1 shows a partitioning of the variables that we use to define the following inference problem:

$$\mathrm{q}(\underline{x}_{\mathrm{J}_0}) = \max_{\underline{x}_{\mathrm{J}_3}} \sum_{\underline{x}_{\mathrm{J}_2}} \min_{\underline{x}_{\mathrm{J}_1}} \min_{\mathrm{I}} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}})$$

We can associate this problem with the following semantics: we may think of each factor as an agent, where $\mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}})$ is the payoff for agent I, which only depends on a subset of variables $\underline{x}_{\mathrm{I}}$. We have adversarial variables $(\underline{x}_{\mathrm{J}_1})$, environmental or chance variables $(\underline{x}_{\mathrm{J}_2})$, controlled variables $(\underline{x}_{\mathrm{J}_3})$ and query variables $(\underline{x}_{\mathrm{J}_0})$. The inference problem above for each query $\underline{x}_{\mathrm{J}_0}$ seeks to maximize the expected minimum payoff of all agents, without observing the adversarial or chance variables, and assuming the adversary makes its decision after observing control and chance variables.

Example 1.1.4. A "probabilistic" graphical model is defined using a expansion semigroup $\mathcal{G}_e = (\mathbb{R}^{\geq 0}, \times)$ and often a marginalization semigroup $\mathcal{G}_m = (\mathbb{R}^{\geq 0}, +)$. The expanded form represents the unnormalized joint probability $\mathrm{q}(\underline{x}) = \prod_{\mathrm{I}} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}})$, whose marginal probabilities are simply called marginals. Replacing the summation with marginalization semigroup $\mathcal{G}_m = (\mathbb{R}^{\geq 0}, \max)$, seeks the maximum probability state and the resulting integration problem $\mathrm{q}(\emptyset) = \max_{\underline{x}} \prod_{\mathrm{I}} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}})$ is known as **maximum a posteriori (MAP)** inference. Alternatively by adding a second marginalization operation to the summation, we get the **marginal MAP** inference

$$\mathrm{q}(\underline{x}_{\mathrm{J}_0}) \quad = \quad \max_{\underline{x}_{\mathrm{J}_2}} \sum_{\underline{x}_{\mathrm{J}_1}} \prod_{\mathrm{I}} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}). \tag{1.4}$$

where here $\bigotimes = \prod$, $\bigoplus^{1} = \sum$ and $\bigoplus^{2} = \max$.

If the object of interest is the negative log-probability (a.k.a. energy), the product expansion semigroup is replaced by $\mathcal{G}_e = (\mathbb{R}, +)$. Instead of sum marginalization semigroup, we can use the **log-sum-exp** semigroup, $\mathcal{G}_m = (\mathbb{R}, +)$ where $a \oplus b \overset{\text{def}}{=} \log(e^{-a} + e^{-b})$. The integral in this case

is the log-partition function. If we change the marginalization semigroup to $\mathscr{G}_m = (\mathbb{R}, \min)$, the integral is the minimum energy (corresponding to MAP).

A well-known example of a probabilistic graphical model is the **Ising model** of ferromagnetism. This model is an extensively studied in physics, mainly to model the phase transition in magnets. The model consists of binary variables ($x_i \in \{-1,1\}$) – denoting magnet spins – arranged on the nodes of a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ (usually a grid or Cayley tree). The energy function (*i.e.*, Hamiltonian) associated with a configuration $\underline{x}$ is the joint form

$$\mathrm{e}(\underline{x}) \quad = \quad \mathrm{q}(\underline{x}) \quad = \quad -\sum_{(i,j)\in\mathcal{E}} x_i \; \mathcal{J}_{ij} \; x_j - \sum_{i\in\mathcal{V}} h_i \tag{1.5}$$

Variable interactions are denoted by $\mathcal{J}$ and h is called the local field. Here each $\mathcal{J}_{i,j}$ defines a factor over x_i, x_j: $\mathrm{f}_{\{i,j\}}(\underline{x}_{\{i,j\}}) = -x_i \; \mathcal{J}_{ij} \; x_j$ and local fields define local factors $\mathrm{f}_{\{i\}}(x_i) = -h_i x_i$.

Depending on the type of interactions, we call the resulting Ising model:

- **ferromagnetic**, if all $\mathcal{J}_{ij} > 0$. In this setting, neighbouring variables are likely to take similar values.
- **anti-ferromagnetic**, if all $\mathcal{J}_{ij} < 0$.
- non-ferromagnetic, if both kind of interactions are allowed. In particular, if the ferromagnetic and anti-ferromagnetic interactions have comparable frequency, the model is called **spin-glass**. This class of problem shows most interesting behaviours, which is not completely understood [212]. As we will see, the studied phenomena in these materials have important connections to difficult inference problems including combinatorial optimization problems. Two well studied models of spin glass are Edward-Anderson (EA [93]) and Sherrington-Kirkpatrick (SK [280]) models. While the EA model is defined on a grid (*i.e.*, spin-glass interactions over a grid), the SK model is a complete graph.

1.2 The inference hierarchy

Often, the complexity class is concerned with the **decision version** of the inference problem in definition 1.1.4. The decision version of an inference problem asks a yes/no question about the integral: $\mathrm{q}(\emptyset) \overset{?}{\geq} q$ for a given q.

Here, we produce a hierarchy of inference problems in analogy to polynomial [288], the counting [302] and arithmetic [264] hierarchies.

To define the hierarchy, we assume the following in definition 1.1.4:

– Any two consecutive marginalization operations are distinct ($\overset{l}{\oplus} \neq \overset{l+1}{\oplus} \; \forall 1 \leq l < M$).

– The marginalization index sets $\mathrm{J}_l \; \forall 1 \leq l \leq M$ are non-empty. Moreover if $|\mathrm{J}_l| = O(\log(N))$ we call this marginalization operation a **polynomial marginalization** as here $|\mathcal{X}_{\mathrm{J}_l}| = \mathrm{Poly}(N)$.

– In defining the factor-graph, we required each factor to be polynomially computable. In

building the hierarchy, we require the operations over each semigroup to be polynomially computable as well. To this end we consider the set of rational numbers $\mathcal{Y}^* \subseteq \mathbb{Q}^{\geq 0} \cup \{\pm\infty\}$. Note that this automatically eliminates semigroups that involve operations such as exponentiation and logarithm (because $\mathbb{Q}$ is not closed under these operations) and only consider summation, product, minimization and maximization.

We can always re-express any inference problem to enforce the first two conditions and therefore they do not impose any restriction. In the following we will use the a **language** to identify inference problems for an arbitrary set of factors $\mathcal{F} = \{\mathrm{f}_\mathrm{I}\}$. For example, sum-product refers to the inference problem $\sum_{\underline{x}} \prod_\mathrm{I} \mathrm{f}_\mathrm{I}(\underline{x}_\mathrm{I}) \overset{?}{\geq} q$. In this sense the rightmost "token" in the language (here *product*) identifies the expansion semigroup $\mathcal{G}_e = (\mathbb{Q}, \prod)$ and the rest of tokens identify the marginalization semigroups over $\mathbb{Q}$ in the given order. Therefore, this minimal language exactly identifies the inference problem. The only information that affects the computational complexity of an inference problem but is not specified in this language is whether each of the marginalization operations are polynomial or exponential.

We define five **inference families**: $\Sigma, \Pi, \Phi, \Psi, \Delta$. The families are associated with that "outermost" marginalization operation – *i.e.*, $\overset{M}{\oplus}$ in definition 1.1.4). Σ is the family of inference problems where $\overset{M}{\oplus}$ = sum. Similarly, Π is associated with product, Φ with minimization and Ψ with maximization. Δ is the family of inference problems where the last marginalization is polynomial (*i.e.*, $|\mathrm{J}_M| = O(\log(N))$ regardless of $\overset{M}{\oplus}$).

Now we define **inference classes** in each family, such that all the problems in the same class have the same computational complexity. Here, the hierarchy is exhaustive – *i.e.*, it includes all inference problems with four operations sum, min, max and product whenever the integral $\mathrm{q}(\emptyset)$ has a polynomial representation (see claim 1.2.1). Moreover the inference classes are disjoint. For this, each family is parameterized by a subscript M and two sets $\mathcal{S}$ and $\mathcal{D}$ (*e.g.*, $\Phi_M(\mathcal{S}, \mathcal{D})$ is an inference "class" in family Φ). As before, M is the number of marginalization operations, $\mathcal{S}$ is the set of indices of the (exponential) sum-marginalization and $\mathcal{D}$ is the set of indices of polynomial marginalizations.

Example 1.2.1. Sum-min-sum-product identifies the decision problem

$$\sum_{\underline{x}_{\mathrm{J}_3}} \min_{\underline{x}_{\mathrm{J}_2}} \sum_{\underline{x}_{\mathrm{J}_1}} \prod_\mathrm{I} \mathrm{f}_\mathrm{I}(\underline{x}_\mathrm{I}) \overset{?}{\geq} q$$

where $\mathrm{J}_1, \mathrm{J}_2$ and J_3 partition $\mathcal{N}$. Assume $\mathrm{J}_1 = \{2, \ldots, \frac{N}{2}\}$, $\mathrm{J}_2 = \{\frac{N}{2} + 1, \ldots, N\}$ and $\mathrm{J}_3 = \{1\}$. Since we have three marginalization operations $M = 3$. Here the first and second marginalizations are exponential and the third one is polynomial (since $|\mathrm{J}_3|$ is constant). Therefore $\mathcal{D} = \{3\}$. Since the only exponential summation is $\overset{1}{\bigoplus}_{\underline{x}_{\mathrm{J}_1}} = \sum_{\underline{x}_{\mathrm{J}_1}}$, $\mathcal{S} = \{1\}$. In our inference hierarchy, this problem belongs to the class $\Delta_3(\{1\}, \{3\})$.

Alternatively, if we use different values for J_1, J_2 and J_3 that all linearly grow with N, the corresponding inference problem becomes a member of $\Sigma_3(\{1,3\},\emptyset)$.

Remark 1. Note that arbitrary assignments to M, $\mathcal{S}$ and $\mathcal{D}$ do not necessarily define a valid inference class. For example we require that $\mathcal{S} \cap \mathcal{D} = \emptyset$ and no index in $\mathcal{D}$ and $\mathcal{S}$ are is larger than M. Moreover, the values in $\mathcal{S}$ and $\mathcal{D}$ should be compatible with the inference class. For example, for inference class $\Sigma_M(\mathcal{S},\mathcal{D})$, M is a member of $\mathcal{S}$. For notational convenience, if an inference class notation is invalid we equate it with an empty set – *e.g.*, $\Psi_1(\{1\},\emptyset) = \emptyset$, because $\mathcal{S} = \{1\}$ and $M = 1$ means the inference class is Σ rather than Ψ.

In the definition below, we ignore the inference problems in which product appears in any of the marginalization semigroups (*e.g.*, product-sum). The following claim, explains this choice.

Claim 1.2.1. *For* $\oplus_M = \text{prod}$*, the inference query* $\mathrm{q}(\underline{x}_{J_0})$ *can have an exponential representation in* N.

Proof. The claim states that when the product appears in the marginalization operations, the marginal (and integral) can become very large, such that we can no longer represent them in polynomial space in N. We show this for an integration problem. The same idea can show the exponential representation of a marginal query.

To see why this integral has an exponential representation in N, consider its simplified form

$$\mathrm{q}(\emptyset) = \prod_{\underline{x}_I} \mathrm{q}(\underline{x}_I)$$

where $\mathrm{q}(\underline{x})$ here is the result of inference up to the last marginalization step $\overset{M}{\oplus}$, which is product, where $\mathcal{X}_I$ grows exponentially with N. Recall that the hierarchy is defined for operations on $\mathbb{Q}^{\geq 0}$. Since $\mathrm{q}(\underline{x}_I)$ for each $\underline{x}_I \in \mathcal{X}_I$ has a constant size, say c, the size of representation of $\mathrm{q}(\emptyset)$ using a binary scheme is

$$\lceil \log_2(\mathrm{q}(\emptyset)) \rceil = \left\lceil \log_2 \Big(\prod_{\underline{x}_I} \mathrm{q}(\underline{x}_I) \Big) \right\rceil = \left\lceil \sum_{\underline{x}_I} c \right\rceil = \lceil c|\mathcal{X}_I| \rceil$$

which is exponential in N. □

Define the **base members** of families as

$$\begin{aligned} \Sigma_0(\emptyset,\emptyset) &\overset{\text{def}}{=} \{\text{sum}\} \quad \Phi_0(\emptyset,\emptyset) \overset{\text{def}}{=} \{\text{min}\} \\ \Psi_0(\emptyset,\emptyset) &\overset{\text{def}}{=} \{\text{max}\} \quad \Pi_0(\emptyset,\emptyset) \overset{\text{def}}{=} \{\text{prod}\} \\ \Delta_0(\emptyset,\emptyset) &= \emptyset \quad \Delta_1(\emptyset,\{1\}) \overset{\text{def}}{=} \{\text{sum}-\text{sum}, \text{min}-\text{min}, \text{max}-\text{max}\} \end{aligned} \tag{1.6}$$

where the initial members of each family only identify the expansion semigroup – *e.g.*, sum in $\Sigma_0(\emptyset,\emptyset)$ identifies $\mathrm{q}(\underline{x}) = \sum_{\mathrm{I}} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}})$. Here, the exception is $\Delta_1(\emptyset,\{1\})$, which contains three *inference problems*.[2]

Let $\Xi_M(\mathcal{S},\mathcal{D})$ denote the union of corresponding classes within all families:

$$\Xi_M(\mathcal{S},\mathcal{D}) = \Sigma_M(\mathcal{S},\mathcal{D}) \cup \Pi_M(\mathcal{S},\mathcal{D}) \cup \Phi_M(\mathcal{S},\mathcal{D}) \cup \Psi_M(\mathcal{S},\mathcal{D}) \cup \Delta_M(\mathcal{S},\mathcal{D})$$

Now define the **inference family members** recursively, by adding a marginalization operation to all the problems in each inference class. If this marginalization is polynomial then the new class belongs to the Δ family and the set $\mathcal{D}$ is updated accordingly. Alternatively, if this outermost marginalization is exponential, depending on the new marginal operation (*i.e.*, $\min, \max, \mathrm{sum}$) the new class is defined to be a member of Φ, Ψ or Σ. For the case that the last marginalization is summation set $\mathcal{S}$ is updated.

- **Adding an exponential marginalization** $\forall\ |\mathcal{X}_{\mathrm{J}_M}| = \mathrm{Poly}(N), M > 0$

$$\begin{aligned}
\Sigma_{M+1}(\mathcal{S} \cup \{M+1\},\mathcal{D}) &\stackrel{\text{def}}{=} \left\{\mathrm{sum} - \xi \mid \xi \in \Xi_M(\mathcal{S},\mathcal{D}) \setminus \Sigma_M(\mathcal{S},\mathcal{D})\right\} \\
\Phi_{M+1}(\mathcal{S},\mathcal{D}) &\stackrel{\text{def}}{=} \left\{\min -\xi \mid \xi \in \Xi_M(\mathcal{S},\mathcal{D}) \setminus \Phi_M(\mathcal{S},\mathcal{D})\right\} \\
\Psi_{M+1}(\mathcal{S},\mathcal{D}) &\stackrel{\text{def}}{=} \left\{\max -\xi \mid \xi \in \Xi_M(\mathcal{S},\mathcal{D}) \setminus \Psi_M(\mathcal{S},\mathcal{D})\right\} \\
\Pi_{M+1}(\mathcal{S},\mathcal{D}) &\stackrel{\text{def}}{=} \emptyset
\end{aligned} \tag{1.7}$$

- **Adding a polynomial marginalization** $\forall\ |\mathcal{X}_{\mathrm{J}_M}| = \mathrm{Poly}(N), M > 1$

$$\Delta_{M+1}(\mathcal{S},\mathcal{D} \cup \{M+1\}) \stackrel{\text{def}}{=} \left\{\oplus -\xi \mid \xi \in \Xi_M(\mathcal{S},\mathcal{D})\ , \oplus \in \{\min,\max,\mathrm{sum}\}\right\} \tag{1.8}$$

1.2.1 Single marginalization

The inference classes in the hierarchy with one marginalization are

$$\Delta_1(\emptyset,\{1\}) = \{\min-\min,\ \max-\max,\ \mathrm{sum}-\mathrm{sum}\} \tag{1.9}$$

$$\Psi_1(\emptyset,\emptyset) = \{\max-\min,\ \max-\mathrm{sum},\ \max-\mathrm{prod}\} \tag{1.10}$$

$$\Phi_1(\emptyset,\emptyset) = \{\min-\max,\ \min-\mathrm{sum},\ \min-\mathrm{prod}\} \tag{1.11}$$

$$\Sigma_1(\{1\},\emptyset) = \{\mathrm{sum}-\mathrm{prod},\ \mathrm{sum}-\min,\ \mathrm{sum}-\max\} \tag{1.12}$$

Now we review all the problems above and prove that Δ_1, Ψ_1, Φ_1 and Σ_1 are complete w.r.t. $\mathbb{P}$, $\mathbb{NP}$, **co**$\mathbb{NP}$ and $\mathbb{PP}$ respectively. Starting from Δ_1:

[2] We treat $M = 1$ for Δ specially as in this case the marginalization operation can not be polynomial. This is because if $|\mathrm{J}_1| = O(\log(N))$, then $|\mathrm{J}_0| = \Omega(N)$ which violates the conditions in the definition of the inference problem.

Proposition 1.2.2. *sum-sum, min-min and max-max inference are in* $\mathbb{P}$.

Proof. To show that these inference problems are in $\mathbb{P}$, we provide polynomial-time algorithms for them:

- sum − sum is short for

$$\mathrm{q}(\emptyset) = \sum_{\underline{x}} \sum_{\mathrm{I}} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}})$$

which asks for the sum over all assignments of $\underline{x} \in \mathcal{X}$, of the sum of all the factors. It is easy to see that each factor value $\mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}})$ $\forall \mathrm{I}$, $\mathcal{X}_{\mathrm{I}}$ is counted $|\mathcal{X}_{\setminus \mathrm{I}}|$ times in the summation above. Therefore we can rewrite the integral above as

$$\mathrm{q}(\emptyset) = \sum_{\mathrm{I}} |\mathcal{X}_{\setminus \mathrm{I}}| \Big(\sum_{\underline{x}_{\mathrm{I}}} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \Big)$$

where the new form involves polynomial number of terms and therefore is easy to calculate.

- min − min (similar for max − max) is short for

$$\mathrm{q}(\emptyset) = \min_{\underline{x}} \min_{\mathrm{I}} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}})$$

where the query seeks the minimum achievable value of any factor. We can easily obtain this by seeking the range of all factors and reporting the minimum value in polynomial time. □

Max-sum and max-prod are widely studied and it is known that their decision version are $\mathbb{NP}$-complete [281]. By reduction from satisfiability we can show that max-min inference [258] is also $\mathbb{NP}$-hard.

Proposition 1.2.3. *The decision version of max-min inference that asks* $\max_{\underline{x}} \min_{\mathrm{I}} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \overset{?}{\geq} q$ *is* $\mathbb{NP}$*-complete.*

Proof. Given $\underline{x}$ it is easy to verify the decision problem, so max-min decision belongs to $\mathbb{NP}$. To show $\mathbb{NP}$-completeness, we reduce the 3-SAT to a max-min inference problem, such that 3-SAT is satisfiable *iff* the max-min value is $\mathrm{q}(\emptyset) \geq 1$ and unsatisfiable otherwise.

Simply define one factor per clause of 3-SAT, such that $\mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) = 1$ if $\underline{x}_{\mathrm{I}}$ satisfies the clause and any number less than one otherwise. With this construction, the max-min value $\max_{\underline{x}} \min_{\mathrm{I} \in \mathcal{F}} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}})$ is one *iff* the original SAT problem was satisfiable, otherwise it is less than one. This reduces 3-SAT to Max-Min-decision. □

This means all the problems in $\Psi_1(\emptyset, \emptyset)$ are in $\mathbb{NP}$ (and in fact are complete w.r.t. this complexity class). In contrast, problems in $\Phi_1(\emptyset, \emptyset)$ are in **co**$\mathbb{NP}$, which is the class of decision problems in which the "NO instances" result has a polynomial time verifiable witness or proof. Note that by

changing the decision problem from $\mathrm{q}(\emptyset) \overset{?}{\geq} q$ to $\mathrm{q}(\emptyset) \overset{?}{\leq} q$, the complexity classes of problems in Φ and Ψ family are reversed (*i.e.*, problems in $\Phi_1(\emptyset,\emptyset)$ become $\mathbb{NP}$-complete and the problems in $\Psi_1(\emptyset,\emptyset)$ become **co**$\mathbb{NP}$-complete).

Among the members of $\Sigma_1(\{1\},\emptyset)$, sum-product is known to be $\mathbb{PP}$-complete [194, 267]. It is easy to show the same result for sum-min (sum-max) inference.

Proposition 1.2.4. *The sum-min decision problem* $\sum_{\underline{x}} \min_{\mathrm{I}} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \overset{?}{\geq} q$ *is* $\mathbb{PP}$*-complete for* $\mathcal{Y} = \{0,1\}$.

$\mathbb{PP}$ is the class of problems that are polynomially solvable using a non-deterministic Turing machine, where the acceptance condition is that the majority of computation paths accept.

Proof. To see that $\sum_{\underline{x}} \min_{\mathrm{I}} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \overset{?}{\geq} q$ is in $\mathbb{PP}$, enumerate all $\underline{x} \in \mathcal{X}$ non-deterministically and for each assignment calculate $\min_{\mathrm{I}} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}})$ in polynomial time (where each path accepts iff $\min_{\mathrm{I}} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) = 1$) and accept iff at least q of the paths accept.

Given a matrix $\mathbf{A} \in \{0,1\}^{N\times N}$ the problem of calculating its permanent

$$\mathrm{perm}(\mathbf{A}) = \sum_{\underline{z}\in\mathcal{S}_N} \prod_{i=1}^{N} \mathbf{A}_{i,z_i}$$

where $\mathcal{S}_N$ is the set of permutations of $1,\ldots,N$ is $\#\mathbb{P}$-complete and the corresponding decision problem is $\mathbb{PP}$-complete [297]. To show completeness w.r.t. $\mathbb{PP}$ it is enough to reduce the problem of computing the matrix permanent to sum-min inference in a graphical model. The problem of computing the permanent has been reduced to sum-product inference in graphical models [139]. However, when $\mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \in \{0,1\}\ \forall \mathrm{I}$, sum-product is isomorphic to sum-min. This is because $y_1 \times y_2 = \min(y_1, y_2) \forall y_i \in \{0,1\}$. Therefore, the problem of computing the permanent for such matrices reduces to sum-min inference in the factor-graph of [139]. □

1.2.2 Complexity of general inference classes

Let $\mho(.)$ denote the complexity class of an inference class in the hierarchy. In obtaining the complexity class of problems with $M > 1$, we use the following fact, which is also used in the polynomial hierarchy: $\mathbb{P}^{\mathbb{NP}} = \mathbb{P}^{\mathbf{co}\mathbb{NP}}$ [14]. In fact $\mathbb{P}^{\mathbb{NP}^{\mathbb{A}}} = \mathbb{P}^{\mathbf{co}\mathbb{NP}^{\mathbb{A}}}$, for any oracle $\mathbb{A}$. This means that by adding a polynomial marginalization to the problems in $\Phi_M(\mathcal{S},\mathcal{D})$ and $\Psi_M(\mathcal{S},\mathcal{D})$, we get the same complexity class $\mho(\Delta_{M+1}(\mathcal{S},\mathcal{D}\cup\{M+1\}))$. The following gives a recursive definition of complexity class for problems in the inference hierarchy.[3] Note that the definition of the complexity for each class is very similar to the recursive definition of members of each class in equations (1.7) and (1.8)

[3] We do not prove the completeness w.r.t. complexity classes beyond the first level of the hierarchy and only assert the membership.

Theorem 1.2.5. *The complexity of inference classes in the hierarchy is given by the recursion*

$$\mathfrak{U}(\Phi_{M+1}(\mathcal{S},\mathcal{D})) = \mathbf{co}\mathbb{NP}^{\mathfrak{U}(\Xi_M(\mathcal{S},\mathcal{D})\setminus\Phi_M(\mathcal{S},\mathcal{D}))} \tag{1.13}$$

$$\mathfrak{U}(\Psi_{M+1}(\mathcal{S},\mathcal{D})) = \mathbb{NP}^{\mathfrak{U}(\Xi_M(\mathcal{S},\mathcal{D})\setminus\Psi_M(\mathcal{S},\mathcal{D}))} \tag{1.14}$$

$$\mathfrak{U}(\Sigma_{M+1}(\mathcal{S}\cup\{M+1\},\mathcal{D})) = \mathbb{PP}^{\mathfrak{U}(\Xi_M(\mathcal{S},\mathcal{D})\setminus\Sigma_M(\mathcal{S},\mathcal{D}))} \tag{1.15}$$

$$\mathfrak{U}(\Delta_{M+1}(\mathcal{S},\mathcal{D}\cup\{M+1\})) = \mathbb{P}^{\mathfrak{U}(\Xi_M(\mathcal{S},\mathcal{D}))} \tag{1.16}$$

where the base members are defined in equation (1.6) *and belong to* $\mathbb{P}$.

Proof. Recall that our definition of factor graph ensures that $\mathrm{q}(\underline{x})$ can be evaluated in polynomial time and therefore the base members are in $\mathbb{P}$ (for complexity of base members of Δ see proposition 1.2.2). We use these classes as the base of our induction and assuming the complexity classes above are correct for M we show that are correct for $M+1$. We consider all the above statements one by one:

- *Complexity for members of* $\Phi_{M+1}(\mathcal{S},\mathcal{D})$:

Adding an exponential-sized *min*-marginalization to an inference problem with known complexity $\mathbb{A}$, requires a Turing machine to non-deterministically enumerate $\underline{z}_{\mathrm{J}_M} \in \mathcal{X}_{\mathrm{J}_M}$ possibilities, then call the $\mathbb{A}$ oracle with the "reduced factor-graph" – in which $\underline{x}_{\mathrm{J}_M}$ is clamped to $\underline{z}_{\mathrm{J}_M}$ – and reject iff any of the calls to oracle rejects. This means $\mathfrak{U}(\Phi_{M+1}(\mathcal{S},\mathcal{D})) = \mathbf{co}\mathbb{NP}^{\mathbb{A}}$.

Here, equation (1.13) is also making another assumption expressed in the following claim.

Claim 1.2.6. *All inference classes in* $\Xi_M(\mathcal{S},\mathcal{D}) \setminus \Phi_M(\mathcal{S},\mathcal{D})$ *have the same complexity* $\mathbb{A}$.

- $M = 0$: the fact that $\mathrm{q}(\underline{x})$ can be evaluated in polynomial time means that $\mathbb{A} = \mathbb{P}$.
- $M > 0$: $\Xi_M(\mathcal{S},\mathcal{D}) \setminus \Phi_M(\mathcal{S},\mathcal{D})$ only contains one inference class – that is exactly only one of the following cases is correct:
  - $M \in \mathcal{S} \;\Rightarrow\; \Xi_M(\mathcal{S},\mathcal{D}) \setminus \Phi_M(\mathcal{S},\mathcal{D}) = \Sigma_M(\mathcal{S},\mathcal{D})$
  - $M \in \mathcal{D} \;\Rightarrow\; \Xi_M(\mathcal{S},\mathcal{D}) \setminus \Phi_M(\mathcal{S},\mathcal{D}) = \Delta_M(\mathcal{S},\mathcal{D})$
  - $M \notin \mathcal{S}\cup\mathcal{D} \;\Rightarrow\; \Xi_M(\mathcal{S},\mathcal{D}) \setminus \Phi_M(\mathcal{S},\mathcal{D}) = \Psi_M(\mathcal{S},\mathcal{D})$.
    (in constructing the hierarchy we assume two consecutive marginalizations are distinct and the current marginalization is a minimization.)

  But if $\Xi_M(\mathcal{S},\mathcal{D}) \setminus \Phi_M(\mathcal{S},\mathcal{D})$ contains a single class, the inductive hypothesis ensures that all problems in $\Xi_M(\mathcal{S},\mathcal{D}) \setminus \Phi_M(\mathcal{S},\mathcal{D})$ have the same complexity class $\mathbb{A}$.

This completes the proof of our claim.

- *Complexity for members of* $\Psi_{M+1}(\mathcal{S},\mathcal{D})$:

Adding an exponential-sized *max*-marginalization to an inference problem with known complexity $\mathbb{A}$, requires a Turing machine to non-deterministically enumerate $\underline{z}_{\mathrm{J}_M} \in \mathcal{X}_{\mathrm{J}_M}$ possibilities, then

call the $\mathbb{A}$ oracle with the reduced factor-graph and accept iff any of the calls to oracle accepts. This means $\mathfrak{U}(\Psi_{M+1}(\mathcal{S},\mathcal{D})) = \mathbb{NP}^{\mathbb{A}}$. Here, an argument similar to that of claim 1.2.6 ensures that $\Xi_M(\mathcal{S},\mathcal{D}) \setminus \Psi_M(\mathcal{S},\mathcal{D})$ in equation (1.14) contains a single inference class.

- *Complexity for members of* $\Sigma_{M+1}(\mathcal{S} \cup \{M+1\},\mathcal{D})$:

Adding an exponential-sized *sum*-marginalization to an inference problem with known complexity $\mathbb{A}$, requires a Turing machine to non-deterministically enumerate $\underline{z}_{\mathrm{J}_M} \in \mathcal{X}_{\mathrm{J}_M}$ possibilities, then call the $\mathbb{A}$ oracle with the reduced factor-graph and accept iff majority of the calls to oracle accepts. This means $\mathfrak{U}(\Psi_{M+1}(\mathcal{S},\mathcal{D})) = \mathbb{PP}^{\mathbb{A}}$.

  - $M = 0$: the fact that $\mathrm{q}(\underline{x})$ can be evaluated in polynomial time means that $\mathbb{A} = \mathbb{P}$.
  - $M > 0$:
    - $M \in \mathcal{D} \;\Rightarrow\; \Xi_M(\mathcal{S},\mathcal{D}) \setminus \Sigma_M(\mathcal{S},\mathcal{D}) = \Delta_M(\mathcal{S},\mathcal{D})$.
    - $M \notin \mathcal{D} \cup \mathcal{S} \;\Rightarrow\; \Xi_M(\mathcal{S},\mathcal{D}) \setminus \Sigma_M(\mathcal{S},\mathcal{D}) = \Psi_M(\mathcal{S},\mathcal{D}) \cup \Phi_M(\mathcal{S},\mathcal{D})$: despite the fact that $\mathbb{A} = \mathfrak{U}(\Psi_M(\mathcal{S},\mathcal{D}))$ is different from $\mathbb{A}' = \mathfrak{U}(\Phi_M(\mathcal{S},\mathcal{D}))$, since $\mathbb{PP}$ *is closed under complement*, which means $\mathbb{PP}^{\mathbb{A}} = \mathbb{PP}^{\mathbb{A}}$ and the recursive definition of complexity equation (1.15) remains correct.

- *Complexity for members of* $\Delta_{M+1}(\mathcal{S},\mathcal{D} \cup \{M+1\})$:

Adding a polynomial-sized marginalization to an inference problem with known complexity $\mathbb{A}$, requires a Turing machine to deterministically enumerate $\underline{z}_{\mathrm{J}_M} \in \mathcal{X}_{\mathrm{J}_M}$ possibilities in polynomial time, and each time call the $\mathbb{A}$ oracle with the reduced factor-graph and accept after some polynomial-time calculation. This means $\mathfrak{U}(\Psi_{M+1}(\mathcal{S},\mathcal{D})) = \mathbb{P}^{\mathbb{A}}$. Here, there are three possibilities:

  - $M = 0$: here again $\mathbb{A} = \mathbb{P}$.
  - $M \in \mathcal{S} \;\Rightarrow\; \Xi_M(\mathcal{S},\mathcal{D}) = \Sigma_M(\mathcal{S},\mathcal{D})$.
  - $M \in \mathcal{D} \;\Rightarrow\; \Xi_M(\mathcal{S},\mathcal{D}) = \Delta_M(\mathcal{S},\mathcal{D})$.
  - $M \notin \mathcal{D} \cup \mathcal{S} \;\Rightarrow\; \Xi_M(\mathcal{S},\mathcal{D}) = \Psi_M(\mathcal{S},\mathcal{D}) \cup \Phi_M(\mathcal{S},\mathcal{D})$, in which case since $\mathbb{PP}^{\mathbb{NP}^{\mathbb{B}}} = \mathbb{PP}^{\mathbf{co}\mathbb{NP}^{\mathbb{B}}}$, the recursive definition of complexity in equation (1.16) remains correct.

□

Example 1.2.2. Consider the marginal-MAP inference of equation (1.4). The decision version of this problem, $\mathrm{q}(\emptyset) \overset{?}{\geq} q$, is a member of $\Psi^2(\{1\},\emptyset)$ which also includes $\max-\mathrm{sum}-\min$ and $\max-\mathrm{sum}-\max$. The complexity of this class according to equation (1.14) is $\mathfrak{U}(\Psi^2(\{1\},\emptyset)) = \mathbb{NP}^{\mathbb{PP}}$. However, marginal-MAP is also known to be "complete" w.r.t. $\mathbb{NP}^{\mathbb{PP}}$ [241]. Now suppose that the max-marginalization over $\underline{x}_{\mathrm{J}_2}$ is polynomial (*e.g.*, $|\mathrm{J}_2|$ is constant). Then marginal-MAP belongs to

$\Delta_2(\{1\},\{2\})$ with complexity $\mathbb{PP}^{\mathbb{PP}}$. This is because a Turing machine can enumerate all $\underline{z}_{\mathrm{J}_2} \in \mathcal{X}_{\mathrm{J}_2}$ in polynomial time and call its $\mathbb{PP}$ oracle to see if

$$\begin{aligned} & \mathrm{q}(\underline{x}_{\mathrm{J}_0} \mid \underline{z}_{\mathrm{J}_2}) \overset{?}{\geq} q \\ \text{where} \quad & \mathrm{q}(\underline{x}_{\mathrm{J}_0} \mid \underline{z}_{\mathrm{J}_2}) = \sum_{\underline{x}_{\mathrm{J}_2}} \prod_{\mathrm{I}} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}\setminus\mathrm{J}_2}, \underline{z}_{\mathrm{I}\cap\mathrm{J}_2}) \end{aligned}$$

and *accept* if any of its calls to oracle accepts, and rejects otherwise. Here, $\mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}\setminus\mathrm{J}_2}, \underline{z}_{\mathrm{I}\cap\mathrm{J}_2})$ is the reduced factor, in which all the variables in $\underline{x}_{\mathrm{J}_2}$ are fixed to $\underline{z}_{\mathrm{J}_2\cap\mathrm{I}}$.

The example above also hints at the rationale behind the recursive definition of complexity class for each inference class in the hierarchy. Consider the inference family Φ:

Here, **Toda's theorem** [296] has an interesting implication w.r.t. the hierarchy. This theorem states that $\mathbb{PP}$ is as hard as the polynomial hierarchy, which means $\min - \max - \min - \ldots - \max$ inference for an arbitrary, but constant, number of min and max operations appears below the sum-product inference in the inference hierarchy.

1.2.3 Complexity of the hierarchy

By restricting the domain $\mathcal{Y}^*$ to $\{0,1\}$, min and max become isomorphic to logical AND ($\wedge$) and OR ($\vee$) respectively, where $1 \cong \textsc{true}, 0 \cong \textsc{false}$. By considering the restriction of the inference hierarchy to these two operations we can express quantified satisfiability (QSAT) as inference in a graphical model, where $\wedge \cong \forall$ and $\vee \cong \exists$. Let each factor $\mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}})$ be a disjunction –*e.g.*, $\mathrm{f}(\underline{x}_{i,j,k}) = x_i \vee \neg x_j \vee \neg x_k$. Then we have

$$\forall_{\underline{x}_{\mathrm{J}_M}} \exists_{\underline{x}_{\mathrm{J}_{M-1}}} \ldots \exists_{\underline{x}_{\mathrm{J}_2}} \forall_{\underline{x}_{\mathrm{J}_1}} \bigwedge_{\mathrm{I}} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \cong \min_{\underline{x}_{\mathrm{J}_M}} \max_{\underline{x}_{\mathrm{J}_{M-1}}} \ldots \max_{\underline{x}_{\mathrm{J}_2}} \min_{\underline{x}_{\mathrm{J}_1}} \min_{\mathrm{I}} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}})$$

By adding the summation operation, we can express the stochastic satisfiability [194] and by generalizing the constraints from disjunctions we can represent any quantified constraint problem (QCP) [36]. QSAT, stochastic SAT and QCPs are all $\mathbb{PSPACE}$-complete, where $\mathbb{PSPACE}$ is the class of problems that can be solved by a (non-deterministic) Turing machine in polynomial space. Therefore if we can show that inference in the inference hierarchy is in $\mathbb{PSPACE}$, it follows that inference hierarchy is in $\mathbb{PSPACE}$-complete as well.

Theorem 1.2.7. *The inference hierarchy is $\mathbb{PSPACE}$-complete.*

Proof. (theorem 1.2.7 on page 21) To prove that a problem is $\mathbb{PSPACE}$-complete, we have to show that 1) it is in $\mathbb{PSPACE}$ and 2) a $\mathbb{PSPACE}$-complete problem reduces to it. We already saw that QSAT, which is $\mathbb{PSPACE}$-complete, reduces to the inference hierarchy. But it is not difficult to

input : $\bigoplus^{M}_{\underline{x}_{\mathrm{J}_M}} \bigoplus^{M-1}_{\underline{x}_{\mathrm{J}_{M-1}}} \cdots \bigoplus^{1}_{\underline{x}_{\mathrm{J}_1}} \bigotimes_{\mathrm{I}} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}})$
output: $\mathrm{q}(\underline{x}_{\mathrm{J}_0})$
for ***each*** $\underline{z}_{\mathrm{J}_0} \in \mathcal{X}_{\mathrm{J}_0}$ **do** // loop over the query domain

  for ***each*** $z_{i_N} \in \mathcal{X}_{i_N}$ **do** // loop over $\mathcal{X}_{i_N}$

    .
    .
    .
    for ***each*** $z_{i_1} \in \mathcal{X}_{i_1}$ **do** // loop over $\mathcal{X}_{i_1}$

      $\mathrm{q}_1(z_{i_1}) := \bigotimes_{\mathrm{I}} \mathrm{f}_{\mathrm{I}}(\underline{z}_{\mathrm{I}})$;
    end
    $\mathrm{q}_{i_2}(z_{i_2}) := \bigoplus^{\jmath(i_1)}_{x_{i_1}} \mathrm{q}_1(x_{i_1})$
    .
    .
    .
    $\mathrm{q}_N(z_{i_N}) := \bigoplus^{\jmath(i_{N-1})}_{x_{i_{N-1}}} \mathrm{q}_{N-1}(x_{i_{N-1}})$
  end
  $\mathrm{q}(\underline{z}_{\mathrm{J}_0}) := \bigoplus^{\jmath(i_N)}_{x_{i_N}} \mathrm{q}_N(x_{i_N})$
end

Algorithm 1: inference in $\mathbb{PSPACE}$

show that inference hierarchy is contained in $\mathbb{PSPACE}$. Let

$$\mathrm{q}(\underline{x}_{\mathrm{J}_0}) = \bigoplus^{M}_{\underline{x}_{\mathrm{J}_M}} \bigoplus^{M-1}_{\underline{x}_{\mathrm{J}_{M-1}}} \cdots \bigoplus^{1}_{\underline{x}_{\mathrm{J}_1}} \bigotimes_{\mathrm{I}} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}})$$

be any inference problem in the hierarchy. We can simply iterate over all values of $\underline{z} \in \mathcal{X}$ in nested loops or using a recursion. Let $\jmath(i) : \{1, \ldots, N\} \to \{1, \ldots, M\}$ be the index of the marginalization that involves x_i – that is $i \in \mathrm{J}_{\jmath(i)}$. Moreover let $i_1, \ldots, i_N$ be an ordering of variable indices such that $\jmath(i_k) \leq \jmath(i_{k+1})$. Algorithm 1 uses this notation to demonstrate this procedure using nested loops. Note that here we loop over individual domains $\mathcal{X}_{i_k}$ rather than $\mathcal{X}_{\mathrm{J}_m}$ and track only temporary tuples q_{i_k}, so that the space complexity remains polynomial in N.

□

1.3 Polynomial-time inference

Our definition of inference was based on an expansion operation $\otimes$ and one or more marginalization operations $\overset{1}{\oplus}, \ldots, \overset{M}{\oplus}$. If we assume only a single marginalization operation, polynomial time inference is still not generally possible. However, if we further assume that the expansion operation is distributive over marginalization and the factor-graph has no loops, exact polynomial time inference is possible.

Definition 1.3.1. A **commutative semiring** $\mathscr{S} = (\mathscr{Y}^*, \oplus, \otimes)$ is the combination of two commutative semigroups $\mathscr{G}_e = (\mathscr{Y}^*, \otimes)$ and $\mathscr{G}_m = (\mathscr{Y}^*, \oplus)$ with two additional properties

- identity elements $\overset{\oplus}{1}$ and $\overset{\otimes}{1}$ such that $\overset{\oplus}{1} \oplus a = a$ and $\overset{\otimes}{1} \otimes a = a$. Moreover $\overset{\oplus}{1}$ is an **annihilator** for $\mathscr{G}_e = (\otimes, \mathscr{Y}^*)$: $a \otimes \overset{\oplus}{1} = \overset{\oplus}{1} \quad \forall a \in \mathscr{Y}^*$.[4]
- distributive property:

$$a \otimes (b \oplus c) = (a \otimes b) \oplus (a \otimes b) \quad \forall a, b, c \in \mathscr{Y}^*$$

The mechanism of efficient inference using distributive law can be seen in a simple example: instead of calculating $\min(a+b, a+c)$, using the fact that summation distributes over minimization, we may instead obtain the same result using $a + \min(b, c)$, which requires fewer operations.

Example 1.3.1. The following are some examples of commutative semirings:

– Sum-product $(\mathbb{R}^{\geq 0}, +, \times)$.
– Max-product $(\mathbb{R}^{\geq 0} \cup \{-\infty\}, \max, \times)$ and $(\{0,1\}, \max, \times)$.
– Min-max $(\mathcal{S}, \min, \max)$ on any ordered set $\mathcal{S}$.
– Min-sum $(\mathbb{R} \cup \{\infty\}, \min, +)$ and $(\{0,1\}, \min, +)$.
– Or-and $(\{\textsc{true}, \textsc{false}\}, \vee, \wedge)$.
– Union-intersection $(2^{\mathcal{S}}, \cup, \cap)$ for any power-set $2^{\mathcal{S}}$.
– The semiring of natural numbers with greatest common divisor and least common multiple $(\mathcal{N}, \text{lcm}, \text{gcd})$.
– Symmetric difference-intersection semiring for any power-set $(2^{\mathcal{S}}, \nabla, \cap)$.

Many of the semirings above are isomorphic –*e.g.*, $y' \cong -\log(y)$ defines an isomorphism between min-sum and max-product. It is also easy to show that the or-and semiring is isomorphic to min-sum/max-product semiring on $\mathscr{Y}^* = \{0,1\}$.

The inference problems in the example above have different properties indirectly inherited from their commutative semirings: for example, the operation min (also max) is a **choice function**, which means $\min_{a \in \mathcal{A}} a \quad \in \mathcal{A}$. The implication is that if sum of the semiring is min (or

[4] That is when dealing with reals, this is $\overset{\oplus}{1} = 0$; this means $a \times 0 = 0$.

max), we can replace it with $\arg_{\underline{x}_J M}$ max and (if required) recover $\mathrm{q}(\emptyset)$ using $\mathrm{q}(\emptyset) = \bigotimes_I \mathrm{f}_I(\underline{x}^*)$ in polynomial time.

As another example, since both operations have inverses, sum-product is a **field** [247]. The availability of inverse for $\otimes$ operation – *i.e.*, when $\mathscr{G}_e$ is an Abelian group – has an important implication for inference: the expanded form of equation (1.1) can be normalized, and we may inquire about **normalized marginals**

$$\mathrm{p}(\underline{x}_J) = \bigoplus_{\underline{x}_{\setminus J}} \mathrm{p}(\underline{x}) \tag{1.17}$$

$$\text{where} \quad \mathrm{p}(\underline{x}) \overset{\text{def}}{=} \frac{1}{\mathrm{q}(\emptyset)} \otimes \Big(\bigotimes_I \mathrm{f}_I(\underline{x}_I) \Big) \qquad \text{if} \quad \mathrm{q}(\emptyset) \neq \overset{\oplus}{1} \tag{1.18}$$

$$\mathrm{p}(\underline{x}) \overset{\text{def}}{=} \overset{\oplus}{1} \quad \text{if} \quad \mathrm{q}(\emptyset) = \overset{\oplus}{1} \tag{1.19}$$

where $\mathrm{p}(\underline{x})$ is the normalized joint form. We deal with the case where the integral evaluates to the annihilator as a special case because division by annihilator may not be well-defined. This also means, when working with normalized expanded form and normalized marginals, we always have $\bigoplus_{\underline{x}_J} \mathrm{p}(\underline{x}_J) = \overset{\otimes}{1}$

Example 1.3.2. Since $\mathscr{G}_e = (\mathbb{R}^{>0}, \times)$ and $\mathscr{G}_e = (\mathbb{R}, +)$ are both Abelian groups, min-sum and sum-product inference have normalized marginals. For min-sum inference this means $\min_{\underline{x}_J} \mathrm{p}(\underline{x}_J) = \overset{\text{sum}}{1} = 0$. However, for min-max inference, since $(\mathcal{S}, \max)$ is not Abelian, normalized marginals are not defined.

We can apply the identity and annihilator of a commutative semiring to define constraints.

Definition 1.3.2. A **constraint** is a factor $\mathrm{f}_I : \mathcal{X}_I \to \{\overset{\otimes}{1}, \overset{\oplus}{1}\}$ whose range is limited to identity and annihilator of the expansion monoid.[5]

Here, $\mathrm{f}_I(\underline{x}) = \overset{\oplus}{1}$ iff $\underline{x}$ is forbidden and $\mathrm{f}_I(\underline{x}) = \overset{\otimes}{1}$ iff it is permissible. A **constraint satisfaction problem** (CSP) is any inference problem on a semiring in which all factors are constraints. Note that this allows definition of the "same" CSP on any commutative semiring. The idea of using different semirings to define CSPs has been studied in the past [33], however its implication about inference on commutative semirings has been ignored.

Theorem 1.3.1. *Inference in any commutative semiring is $\mathbb{NP}$-hard under randomized polynomial-time reduction.*

Proof. To prove that inference in any semiring $\mathscr{S} = (\mathcal{Y}^*, \overset{\oplus}{1}, \overset{\otimes}{1})$ is $\mathbb{NP}$-hard under randomized polynomial reduction, we deterministically reduce *unique satisfiability* (USAT) to an inference problems on any semiring. USAT is a so-called "promise problem", that asks whether a satisfiability

[5] Recall that a monoid is a semigroup with an identity. The existence of identity here is a property of the semiring.

problem that is promised to have either zero or one satisfying assignment is satisfiable. Valiant and Vazirani [298] prove that a polynomial time randomized algorithm ($\mathbb{RP}$) for USAT implies a $\mathbb{RP}$=$\mathbb{NP}$.

For this reduction consider a set of binary variables $\underline{x} \in \{0,1\}^N$, one per each variable in the given instance of USAT. For each clause, define a constraint factor f_I such that $f_I(\underline{x}_I) = \overset{\otimes}{1}$ if $\underline{x}_I$ satisfies that clause and $f_I(\underline{x}_I) = \overset{\oplus}{1}$ otherwise. This means, $\underline{x}$ is a satisfying assignment for USAT iff $q(\underline{x}) = \bigotimes_I f_I(\underline{x}_I) = \overset{\otimes}{1}$. If the instance is unsatisfiable, the integral $q(\emptyset) = \bigoplus_{\underline{x}} \overset{\oplus}{1} = \overset{\oplus}{1}$ (by definition of $\overset{\oplus}{1}$). If the instance is satisfiable there is only a single instance $\underline{x}^*$ for which $q(\underline{x}^*) = \overset{\otimes}{1}$, and therefore the integral evaluates to $\overset{\otimes}{1}$. Therefore we can decide the satisfiability of USAT by performing inference on any semiring, by only relying on the properties of identities. The satisfying assignment can be recovered using a decimation procedure, assuming access to an oracle for inference on the semiring.

□

Example 1.3.3. Inference on xor-and semiring $(\{\text{TRUE}, \text{FALSE}\}, \text{xor}, \wedge)$, where each factor has a disjunction form, is called parity-SAT, which asks whether the number of SAT solutions is even or odd. A corollary to theorem 1.3.1 is that parity-SAT is $\mathbb{NP}$-hard under randomized reduction, which is indeed the case [298].

We find it useful to use the same notation for the **identity function** 1(condition):

$$1(\text{cond.}) \overset{\text{def}}{=} \begin{cases} & (+,\times) & (\min,+) & (\min,\max) \\ \text{cond.} = \text{TRUE} & 1 & 0 & -\infty \\ \text{cond.} = \text{FALSE} & 0 & +\infty & +\infty \end{cases} \tag{1.20}$$

where the intended semiring for 1(.) function will be clear from the context.

1.4 Reductions

Several of the inference problems over commutative semirings are reducible to each other. Section 1.4.1 reviews the well-known reduction of marginalization to integration for general commutative semirings. We use this reduction to obtain approximate message dependencies in performing loop corrections in section 2.4.

In section 1.4.1, we introduce a procedure to reduce integration to that of finding normalized marginals. The same procedure, called **decimation**, reduces sampling to marginalization. The problem of **sampling** from a distribution is known to be almost as difficult as sum-product integration [151]. As we will see in chapter 3, constraint satisfaction can be reduced to sampling and

therefore marginalization. In section 2.6.3 we introduce a perturbed message passing scheme to perform approximate sampling and use it to solve CSPs. Some recent work perform approximate sampling by finding the MAP solution in the perturbed factor-graph, in which a particular type of noise is added to the factors [125, 239]. Approximate sum-product integration has also been recently reduced to MAP inference [96, 97]. In section 2.3, we see that min-max and min-sum inference can be obtained as limiting cases of min-sum and sum-product inference respectively.

Section 1.4.2 reduces the min-max inference to min-sum also to a sequence of CSPs (and therefore sum-product inference) over factor-graphs. This reduction gives us a powerful procedure to solve min-max problems, which we use in part II to solve bottleneck combinatorial problems.

In contrast to this type of reduction between various modes of inference, many have studied reductions of different types of factor-graphs [90]. Some examples of these special forms are factor-graphs with: binary variables, pairwise interactions, constant degree nodes, and planar form. For example Sanghavi et al. [274] show that min-sum integration is reducible to *maximum independent-set* problem. However since a pairwise binary factor-graph can represent a maximum independent-set problem (see section 3.7), this means that min-sum integration in any factor-graph can be reduced to the same problem on a pairwise binary model.

These reductions are in part motivated by the fact that under some further restrictions the restricted factor-graph allows more efficient inference. For example, (I) it is possible to calculate the sum-product integral of the *planar* spin-glass Ising model (see example 1.1.4) in polynomial time, in the absence of local fields [99]; (II) the complexity of the loop correction method that we study in section 2.4.2 grows exponentially with the degree of each node and therefore it may be beneficial to consider reduced factor-graph where $|\partial i| = 3$; and (III) if the factors in a factor-graphs with pairwise factors satisfy certain metric property, polynomial algorithms can obtain the exact min-sum integral using graph-cuts [41].

1.4.1 Marginalization and integration

This section shows how for arbitrary commutative semirings there is a reduction from marginalization to integration and vice versa.

Marginalization reduces to integration

For any fixed assignment to a subset of variables $\underline{x}_A = \underline{z}_A$ (a.k.a. **evidence**), we can reduce all the factors $f_I(\underline{x}_I)$ that have non-empty intersection with A (*i.e.*, $I \cap A \neq \emptyset$) accordingly:

$$f_{I\setminus A}(\underline{x}_{I\setminus A} \mid \underline{z}_A) \stackrel{\text{def}}{=} \bigoplus_{\underline{x}_{I\cap A}} f_I(\underline{x}_I) \otimes 1(\underline{x}_{I\cap A} = \underline{z}_{I\cap A}) \quad \forall I \; s.t. \; A \cap I \neq \emptyset \tag{1.21}$$

where the identity function $1(.)$ is defined by equation (1.20). The new factor graph produced by **clamping** all factors in this manner, has effectively accounted for the evidence. Marginalization or integration, can be performed on this reduced factor-graph. We use similar notation for the *integral and marginal in the new factor graph – i.e.*, $\mathrm{q}(\emptyset \mid \underline{x}_\mathrm{A})$ and $\mathrm{q}(\underline{x}_\mathrm{B} \mid \underline{x}_\mathrm{A})$. Recall that the problem of integration is that of calculating $\mathrm{q}(\emptyset)$. We can obtain the marginals $\mathrm{q}(\underline{z}_\mathrm{A})$ by integration on reduced factor-graphs for all $\underline{z}_\mathrm{A} \in \mathcal{X}_\mathrm{A}$ reductions.

Claim 1.4.1.

$$\mathrm{q}(\underline{z}_\mathrm{A}) \quad = \quad \mathrm{q}(\emptyset \mid \underline{z}_\mathrm{A}) \tag{1.22}$$

Proof.

$$\begin{aligned}
\mathrm{q}(\underline{z}_\mathrm{A}) &= \bigoplus_{\underline{x}_{\setminus\mathrm{A}}} \bigotimes_{\mathrm{I}} \mathrm{f}_\mathrm{I}(\underline{x}_{\mathrm{I}\setminus\mathrm{A}}, \underline{z}_{\mathrm{A}\cap\mathrm{I}}) \\
&= \bigoplus_{\underline{x}} \left(1(\underline{x}_\mathrm{A} = \underline{z}_\mathrm{A}) \otimes \bigotimes_{\mathrm{I}} \mathrm{f}_\mathrm{I}(\underline{x}_{\mathrm{I}\setminus\mathrm{A}}, \underline{z}_{\mathrm{A}\cap\mathrm{I}}) \right) \\
&= \bigoplus_{\underline{x}} \bigotimes_{\mathrm{I}} \left(\mathrm{f}_\mathrm{A}(\underline{x}_\mathrm{A}) \otimes 1(\underline{x}_{\mathrm{I}\cap\mathrm{A}} = \underline{z}_{\mathrm{I}\cap\mathrm{A}}) \right) \\
&= \bigoplus_{\underline{x}} \bigotimes_{\mathrm{I}} \mathrm{f}_{\mathrm{I}\setminus\mathrm{A}}(\underline{x}_{\mathrm{I}\setminus\mathrm{A}} \mid \underline{z}_\mathrm{A}) \quad = \quad \mathrm{q}(\emptyset \mid \underline{z}_\mathrm{A})
\end{aligned}$$

□

where we can then normalize $\mathrm{q}(\underline{x}_\mathrm{I})$ values to get $\mathrm{p}(\underline{x}_\mathrm{I})$ (as defined in equation (1.17)).

Integration reduces to marginalization

Assume we have access to an oracle that can produce the normalized marginals of equation (1.17). We show how to calculate $\mathrm{q}(\emptyset)$ by making N calls to the oracle. Note that if the marginals are not normalized, the integral is trivially given by $\mathrm{q}(\emptyset) = \bigoplus_{\underline{x}_\mathrm{J}} \mathrm{q}(\underline{x}_\mathrm{J})$

Start with $t = 1$, $\mathrm{B}(t = 0) = \emptyset$ and given the normalized marginal over a variable $\mathrm{p}(x_{i(t)})$, fix the $x_{i(t)}$ to an arbitrary value $z_{i(t)} \in \mathcal{X}_{i(t)}$. Then reduce all factors according to equation (1.21). Repeat this process of marginalization and clamping N times until all the variables are fixed. At each point, $\mathrm{B}(t)$ denotes the subset of variables fixed up to step t (including $i(t)$) and $\mathrm{p}(x_{i(t)} \mid \underline{z}_{\mathrm{B}(t-1)}) = \frac{\mathrm{q}(x_{i(t)} \mid \underline{z}_{\mathrm{B}(t-1)})}{\mathrm{q}(\emptyset \mid \underline{z}_{\mathrm{B}(t-1)})}$ refers to the new marginal. Note that we require $i(t) \notin \mathrm{B}(t-1)$ – that is at each step we fix a different variable.

We call an assignment to $x_{i(t)} = z_{i(t)}$ *invalid*, if $\mathrm{p}(z_{i(t)} \mid \underline{z}_{\mathrm{B}(t)}) = \overset{\oplus}{1}$. This is because $\overset{\oplus}{1}$ is the annihilator of the semiring and we want to avoid division by the annihilator. Using equations (1.17) to (1.19), it is easy to show that if $\mathrm{q}(\emptyset) \neq \overset{\oplus}{1}$, a valid assignment always exists (this is because

$\bigoplus_{x_{i(t)}} \mathrm{p}(x_{i(t)} \mid \underline{z}_{\mathrm{B}(t-1)}) = \overset{\otimes}{1}$). Therefore if we are unable to find a valid assignment, it means $\mathrm{q}(\emptyset) = \overset{\oplus}{1}$.

Let $\underline{z} = \underline{z}_{\mathrm{B}(N+1)}$ denote the final joint assignment produced using the procedure above.

Proposition 1.4.2. *The integral in the original factor-graph is given by*

$$\mathrm{q}(\emptyset) \quad = \quad \left(\bigotimes_{\mathrm{I}} \mathrm{f}_{\mathrm{I}}(\underline{z}_{\mathrm{I}}) \right) \otimes \left(\bigotimes_{1 \le t \le N} \mathrm{p}(z_{i(t)} \mid \underline{z}_{\mathrm{B}(t-1)}) \right)^{-1} \tag{1.23}$$

where the inverse is defined according to $\otimes$-operation.

Proof. First, we derive the an equation for "conditional normalized marginals" for semirings where $\otimes$ defines an inverse.

Claim 1.4.3. *For any semiring with normalized joint form we have*

$$\mathrm{p}(\underline{x}) = \mathrm{p}(x_i) \otimes \mathrm{p}(\underline{x}_{\setminus i} \mid x_i)$$

where $\mathrm{p}(\underline{x}_{\setminus i} \mid x_i) = \frac{\mathrm{q}(\underline{x}_{\setminus i} | x_i)}{\mathrm{q}(\emptyset | x_i)}$

To arrive at this equality first note that since $\underline{x} = \underline{x}_{\setminus i}, x_i$, $\mathrm{q}(\underline{x}_{\setminus i} \mid x_i) = \mathrm{q}(\underline{x})$. Then multiply both sides by $\mathrm{q}(x_i) = \mathrm{q}(\emptyset \mid x_i)$ (see claim 1.4.1) to get

$$\begin{aligned} \mathrm{q}(\underline{x}) \otimes \mathrm{q}(\emptyset \mid x_i) &= \mathrm{q}(x_i) \otimes \mathrm{q}(\underline{x}_{\setminus i} \mid x_i) && \Rightarrow \\ \frac{\mathrm{q}(\underline{x})}{\mathrm{q}(\emptyset)} &= \frac{\mathrm{q}(x_i)}{\mathrm{q}(\emptyset)} \otimes \frac{\mathrm{q}(\underline{x}_{\setminus i} \mid x_i)}{\mathrm{q}(\emptyset \mid x_i)} && \Rightarrow \\ \mathrm{p}(\underline{x}) &= \mathrm{p}(x_i) \otimes \mathrm{p}(\underline{x}_{\setminus i} \mid x_i) \end{aligned}$$

where we divided both sides by $\mathrm{q}(\emptyset)$ and moved a term from left to right in the second step.

Now we can apply this repeatedly to get a chain rule for the semiring:

$$\mathrm{p}(\underline{x}) \quad = \quad \mathrm{p}(x_{i_1}) \otimes \mathrm{p}(x_{i_2} \mid x_{i_1}) \otimes \mathrm{p}(x_{i_3} \mid \underline{x}_{\{i_1,i_2\}}) \otimes \ldots \otimes \mathrm{p}(x_{i_N} \mid \underline{x}_{\{i_1,\ldots,i_{N-1}\}})$$

which is equivalent to

$$\mathrm{p}(\underline{x}) \quad = \quad \mathrm{p}(x_{i(1)}) \otimes \mathrm{p}(x_{i(2)} \mid \underline{x}_{\mathrm{B}(1)}) \otimes \ldots \otimes \mathrm{p}(x_{i(N)} \mid \underline{x}_{\mathrm{B}(N-1)}) \quad = \quad \bigotimes_{1 \le t \le N} \mathrm{p}(\underline{x}_{i(t)} \mid \underline{x}_{\mathrm{B}(t-1)})$$

Simply substituting this into definition of $\mathrm{p}(\underline{x})$ (equation (1.18)) and re-arranging we get

$$\bigotimes_{1\le t\le N} \mathrm{p}(\underline{x}_{i(t)} \mid \underline{x}_{\mathrm{B}(t-1)}) \quad = \frac{1}{\mathrm{q}(\emptyset)} \bigotimes_{\mathrm{I}\in\mathcal{F}} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \quad \Rightarrow$$

$$\mathrm{q}(\emptyset) \quad = \quad \left(\bigotimes_{\mathrm{I}} \mathrm{f}_{\mathrm{I}}(\underline{z}_{\mathrm{I}})\right) \otimes \left(\bigotimes_{1\le t\le N} \mathrm{p}(z_{i(t)} \mid \underline{z}_{\mathrm{B}(t-1)})\right)^{-1}$$

□

The procedure of incremental clamping is known as **decimation**, and its variations are typically used for two objectives: (I) recovering the MAP assignment from (max) marginals (assuming a max-product semiring). Here instead of an arbitrary $\underline{z}_{\mathrm{J}} \in \mathcal{X}_{\mathrm{J}}$, one picks $\underline{z}_{\mathrm{J}} = \arg_{\underline{x}_{\mathrm{J}}} \max \ \mathrm{p}(\underline{x}_{\mathrm{J}})$. (II) producing an unbiased sample from a distribution $\mathrm{p}(.)$ (*i.e.*, assuming sum-product semiring). For this we sample from $\mathrm{p}(\underline{x}_{\mathrm{I}})$: $\underline{z}_{\mathrm{J}} \sim \mathrm{p}(\underline{x}_{\mathrm{J}})$.

1.4.2 Min-max reductions

The min-max objective appears in various fields, particularly in building robust models under uncertain and adversarial settings. In the context of probabilistic graphical models, several min-max objectives different from inference in min-max semiring have been previously studied [140, 162] (also see section 2.1.2). In combinatorial optimization, min-max may refer to the relation between maximization and minimization in dual combinatorial objectives and their corresponding linear programs [276], or it may refer to min-max settings due to uncertainty in the problem specification [5, 16].

In part II we will see that several problems that are studied under the class of **bottleneck problems** can be formulated using the min-max semiring. Instances of these problems include bottleneck traveling salesman problem [242], K-clustering [119], K-center problem [87, 164] and bottleneck assignment problem [121].

Edmonds and Fulkerson [92] introduce a bottleneck framework with a duality theorem that relates the min-max objective in one problem instance to a max-min objective in a dual problem. An intuitive example is the duality between the min-max cut separating nodes a and b – the cut with the minimum of the maximum weight – and min-max path between a and b, which is the path with the minimum of the maximum weight [104]. Hochbaum and Shmoys [136] leverages the triangle inequality in metric spaces to find constant factor approximations to several $\mathbb{NP}$-hard min-max problems under a unified framework.

The common theme in a majority of heuristics for min-max or bottleneck problems is the relation of the min-max objective to a CSP [136, 237]. We establish a similar relation within the context of factor-graphs, by reducing the min-max inference problem on the original factor-graph to inference over a CSP factor-graph (see section 1.3) on the reduced factor-graph in section 1.4.2.

In particular, since we use sum-product inference to solve the resulting CSP, we call this reduction, sum-product reduction of min-max inference.

Min-max reduces to min-sum

Here, we show that min-max inference reduces to min-sum, although in contrast to the sum-product reduction of the next subsection, this is not a polynomial time reduction. First, we make a simple observation about min-max inference. Let $\mathcal{Y} = \bigcup_{I \in \mathcal{F}} \mathcal{Y}_I$ denotes the union over the range of all factors. The min-max value belongs to this set $\max_{I \in \mathcal{F}} f_I(x_I^*) \in \mathcal{Y}$. In fact for any assignment $\underline{x}$, $\max_{I \in \mathcal{F}} f_I(\underline{x}_I) \in \mathcal{Y}$.

Now we show how to manipulate the factors in the original factor-graph to produce new factors over the same domain such that the min-max inference on the former corresponds to the min-sum inference on the later.

Lemma 1.4.4. *Any two sets of factors,* $\{f_I\}$ *and* $\{g_I\}$*, over the identical domains* $\{\mathcal{X}_I\}$ *have identical min-max solutions*

$$\arg_{\underline{x}} \min \max_I f_I(\underline{x}_I) = \arg_{\underline{x}} \min \max_I g_I(\underline{x}_I)$$

if $\forall I, J \in \mathcal{F}, \underline{x}_I \in \mathcal{X}_I, \underline{x}_J \in \mathcal{X}_J$

$$f_I(\underline{x}_I) < f_J(\underline{x}_J) \quad \Leftrightarrow \quad g_I(\underline{x}_I) < g_J(\underline{x}_J)$$

Proof. Assume they have different min-max assignments[6] –*i.e.*, $\underline{x}^* = \arg_{\underline{x}} \min \max_I f_I(\underline{x}_I)$, $\underline{x}'^* = \arg_{\underline{x}} \min \max_I g_I(\underline{x}_I)$ and $x^* \neq x'^*$. Let y^* and y'^* denote the corresponding min-max values.

Claim 1.4.5.

$$y^* > \max_I f_I(\underline{x}_I'^*) \quad \Leftrightarrow \quad y'^* < \max_I g_I(\underline{x}_I^*)$$
$$y^* < \max_I f_I(\underline{x}_I'^*) \quad \Leftrightarrow \quad y'^* > \max_I f_I'(\underline{x}_I^*)$$

This simply follows from the condition of the Lemma. But in each case above, one of the assignments y^* or y'^* is not an optimal min-max assignment as there is an alternative assignment that has a lower maximum over all factors. □

This lemma simply states that what matters in the min-max solution is the *relative ordering* in the factor-values.

[6] For simplicity, we are assuming each instance has a single min-max assignment. In case of multiple assignments there is a one-to-one correspondence between them. Here the proof instead starts with the assumption that there is an assignment $\underline{x}^*$ for the first factor-graph that is different from all min-max assignments in the second factor-graph.

Let $y[1] \leq \ldots \leq y[|\mathcal{Y}|]$ be an ordering of elements in $\mathcal{Y}$, and let $\mathrm{r}(f_I(x_I))$ denote the rank in $\{1, \ldots, |\mathcal{Y}|\}$ of $y_\mathrm{I} = \mathrm{f}_\mathrm{I}(\underline{x}_\mathrm{I})$ in this ordering. Define the min-sum reduction of $\{\mathrm{f}_\mathrm{I}\}_{\mathrm{I} \in \mathcal{F}}$ as

$$\mathrm{g}_\mathrm{I}(\underline{x}_\mathrm{I}) = 2^{\mathrm{r}(\mathrm{f}_\mathrm{I}(\underline{x}_\mathrm{I}))} \quad \forall \mathrm{I} \in \mathcal{F}$$

Theorem 1.4.6.

$$\arg_{\underline{x}} \min \sum_{\mathrm{I}} \mathrm{g}_\mathrm{I}(\underline{x}_\mathrm{I}) \quad = \quad \arg_{\underline{x}} \min \max_{\mathrm{I}} \mathrm{f}_\mathrm{I}(\underline{x}_\mathrm{I})$$

where $\{\mathrm{g}_\mathrm{I}\}_\mathrm{I}$ *is the min-sum reduction of* $\{\mathrm{f}_\mathrm{I}\}_\mathrm{I}$.

Proof. First note that since $g(z) = 2^z$ is a monotonically increasing function, the rank of elements in the range of $\{\mathrm{g}_\mathrm{I}\}_\mathrm{I}$ is the same as their rank in the range of $\{\mathrm{f}_\mathrm{I}\}_\mathrm{I}$. Using Lemma 1.4.4, this means

$$\arg_{\underline{x}} \min \max_{\mathrm{I}} \mathrm{g}_\mathrm{I}(\underline{x}_\mathrm{I}) \quad = \quad \arg_{\underline{x}} \min \max_{\mathrm{I}} \mathrm{f}_\mathrm{I}(\underline{x}_\mathrm{I}). \tag{1.24}$$

Since $2^z > \sum_{l=0}^{z-1} 2^l$, by definition of $\{\mathrm{g}_\mathrm{I}\}$ we have

$$\max_{\mathrm{I} \in \mathcal{F}} \mathrm{g}_\mathrm{I}(\underline{x}_\mathrm{I}) > \sum_{\mathrm{I} \in \mathcal{F} \setminus \mathrm{I}^*} \mathrm{g}_\mathrm{I}(\underline{x}_\mathrm{I}) \quad \text{where} \quad \mathrm{I}^* = \arg_\mathrm{I} \max \mathrm{g}_\mathrm{I}(\underline{x}_\mathrm{I})$$

It follows that for $\underline{x}^1, \underline{x}^2 \in \mathcal{X}$,

$$\max_{\mathrm{I}} \mathrm{g}_\mathrm{I}(\underline{x}^1_\mathrm{I}) < \max_{\mathrm{I}} \mathrm{g}_\mathrm{I}(\underline{x}^2_\mathrm{I}) \quad \Leftrightarrow \quad \sum_{\mathrm{I}} \mathrm{g}_\mathrm{I}(\underline{x}^1_\mathrm{I}) < \sum_{\mathrm{I}} \mathrm{g}_\mathrm{I}(\underline{x}^2_\mathrm{I})$$

Therefore

$$\arg_{\underline{x}} \min \max_{\mathrm{I}} \mathrm{g}_\mathrm{I}(\underline{x}_\mathrm{I}) \quad = \quad \arg_{\underline{x}} \min \sum_{\mathrm{I}} \mathrm{g}_\mathrm{I}(\underline{x}_\mathrm{I}).$$

This equality, combined with equation (1.24), prove the statement of the theorem. □

An alternative approach is to use an inverse temperature parameter β and re-state the min-max objective as the min-sum objective at the low temperature limit

$$\lim_{\beta \to +\infty} \arg_{\underline{x}} \min \sum_{\mathrm{I}} \mathrm{f}_\mathrm{I}^\beta(\underline{x}_\mathrm{I}) \quad = \quad \arg_{\underline{x}} \min \max_{\mathrm{I}} \mathrm{f}_\mathrm{I}(\underline{x}_\mathrm{I}) \tag{1.25}$$

Min-max reduces to sum-product

Recall that $\mathcal{Y} = \bigcup_{\mathrm{I} \in \mathcal{F}} \mathcal{Y}_\mathrm{I}$ denote the union over the range of all factors. For any $y \in \mathcal{Y}$, we *reduce* the original min-max problem to a CSP using the following reduction.

Definition 1.4.1. For any $y \in \mathcal{Y}$, p_y**-reduction** of the min-max problem:

$$\underline{x}^* \quad = \quad \arg_{\underline{x}} \min \max_{\mathrm{I} \in \mathcal{F}} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \tag{1.26}$$

is given by

$$\mathrm{p}_y(\underline{x}) \quad \stackrel{\text{def}}{=} \quad \frac{1}{\mathrm{q}_y(\emptyset)} \prod_{\mathrm{I} \in \mathcal{F}} \mathbb{1}(\mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \leq y) \tag{1.27}$$

where $\mathrm{q}_y(\emptyset)$ is the normalizing constant.[7]

This distribution defines a CSP over $\mathcal{X}$, where $\mathrm{p}_y(\underline{x}) > 0$ *iff* $\underline{x}$ is a satisfying assignment. Moreover, $\mathrm{q}_y(\emptyset)$ gives the number of satisfying assignments. The following theorem is the basis of our reduction.

Theorem 1.4.7. *Let $\underline{x}^*$ denote the min-max solution and y^* be its corresponding value* –i.e., $y^* = \max_{\mathrm{I}} \mathrm{f}_{\mathrm{I}}(\underline{x}^*_{\mathrm{I}})$. *Then $\mathrm{p}_y(\underline{x})$ is satisfiable for all $y \geq y^*$ (in particular $\mathrm{p}_y(\underline{x}^*) > 0$) and unsatisfiable for all $y < y^*$.*

Proof. (A) p_y for $y \geq y^*$ is satisfiable: It is enough to show that for any $y \geq y^*$, $\mathrm{p}_y(\underline{x}^*) > 0$. But since

$$\mathrm{p}_y(\underline{x}^*) \quad = \quad \frac{1}{\mathrm{q}_y(\emptyset)} \prod_{\mathrm{I}} \mathbb{1}(\mathrm{f}_{\mathrm{I}}(\underline{x}^*_{\mathrm{I}}) \leq y)$$

and $\mathrm{f}_{\mathrm{I}}(\underline{x}^*_{\mathrm{I}}) \leq y^* \leq y$, all the indicator functions on the rhs evaluate to 1, showing that $\mathrm{p}_y(\underline{x}^*) > 0$.

(B) p_y for $y < y^*$ is not satisfiable: Towards a contradiction assume that for some $\underline{y} < y^*$, $\mathrm{p}_{\underline{y}}$ is satisfiable. Let $\underline{\underline{x}}$ denote a satisfying assignment –*i.e.*, $\mathrm{p}_{\underline{y}}(\underline{\underline{x}}) > 0$. Using the definition of p_y-reduction, this implies that $\mathbb{1}(\mathrm{f}_{\mathrm{I}}(\underline{\underline{x}}_{\mathrm{I}}) \leq \underline{y}) > 0$ for all $\mathrm{I} \in \mathcal{F}$. However this means that $\max_{\mathrm{I}} \mathrm{f}_{\mathrm{I}}(\underline{\underline{x}}_{\mathrm{I}}) \leq \underline{y} < y^*$, which means y^* is not the min-max value. □

This theorem enables us to find a min-max assignment by solving a sequence of CSPs. Let $y[1] \leq \ldots \leq y[|\mathcal{Y}|]$ be an ordering of $y \in \mathcal{Y}$. Starting from $y = y[\lceil \mathrm{N}/2 \rceil]$, if p_y is satisfiable then $y^* \leq y$. On the other hand, if p_y is not satisfiable, $y^* > y$. Using **binary search**, we need to solve $\log(|\mathcal{Y}|)$ CSPs to find the min-max solution. Moreover at any time-step during the search, we have both upper and lower bounds on the optimal solution. That is $\underline{y} < y^* \leq \overline{y}$, where $\mathrm{p}_{\underline{y}}$ is the latest unsatisfiable and $\mathrm{p}_{\overline{y}}$ is the latest satisfiable reduction.

However, finding an assignment $\underline{x}^*$ such that $\mathrm{p}_y(\underline{x}^*) > 0$ or otherwise showing that no such assignment exists, is in general, $\mathbb{NP}$-hard. Instead, we can use an incomplete solver [160], which may find a solution if the CSP is satisfiable, but its failure to find a solution does not guarantee

[7] To always have a well-defined probability, we define $\frac{0}{0} \stackrel{\text{def}}{=} 0$.

unsatisfiability. By using an incomplete solver, we lose the lower bound $\underline{y}$ on the optimal min-max solution.[8] However the following theorem states that, as we increase y from the min-max value y^*, the number of satisfying assignments to p_y-reduction increases, making it potentially easier to solve.

Proposition 1.4.8.

$$y_1 < y_2 \quad \Rightarrow \quad \mathrm{q}_{y_1}(\emptyset) \leq \mathrm{q}_{y_2}(\emptyset) \qquad \forall y_1, y_2 \in \mathcal{Y}$$

where $\mathrm{q}_y(\emptyset)$ (i.e., *partition function) is the number of solutions of* p_y*-reduction.*

Proof. Recall the definition $\mathrm{q}_y(\emptyset) = \sum_{\underline{x}} \prod_{\mathrm{I}} 1(\mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \leq y)$. For $y_1 < y_2$ we have:

$$\begin{aligned}
&\mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \leq y_1 \quad \rightarrow \quad \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \leq y_2 && \Rightarrow \\
&1(\mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \leq y_1) \leq 1(\mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \leq y_2) && \Rightarrow \\
&\sum_{\underline{x}} \prod_{\mathrm{I}} 1(\mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \leq y_1) \leq \sum_{\underline{x}} \prod_{\mathrm{I}} 1(\mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \leq y_2) && \Rightarrow \\
&\mathrm{q}_{y_1}(\emptyset) \leq \mathrm{q}_{y_2}(\emptyset)
\end{aligned}$$

□

This means that the sub-optimality of our solution is related to our ability to solve CSP-reductions – that is, as the gap $y - y^*$ increases, the p_y-reduction potentially becomes easier to solve.

[8] To maintain the lower bound one should be able to correctly assert unsatisfiability.

Chapter 2

Approximate inference

2.1 Belief Propagation

A naive approach to inference over commutative semirings

$$\mathrm{q}(\underline{x}_{\mathrm{J}}) \quad = \quad \bigoplus_{\underline{x}_{\setminus \mathrm{J}}} \bigotimes_{\mathrm{I}} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \tag{2.1}$$

or its normalized version (equation (1.17)), is to construct a complete N-dimensional array of $\mathrm{q}(\underline{x})$ using the tensor product $\mathrm{q}(\underline{x}) \;=\; \bigotimes_{\mathrm{I}} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}})$ and then perform $\oplus$-marginalization. However, the number of elements in $\mathrm{q}(\underline{x})$ is $|\mathcal{X}|$, which is exponential in N, the number of variables.

If the factor-graph is loop free, we can use distributive law to make inference tractable. Assuming $\mathrm{q}(\underline{x}_{\mathrm{K}})$ (or $\mathrm{q}(x_k)$) is the marginal of interest, form a tree with K (or k) as its root. Then starting from the leaves, using the distributive law, we can move the $\oplus$ inside the $\otimes$ and define "messages" from leaves towards the root as follows:

$$\mathrm{q}_{i \to \mathrm{I}}(x_i) \quad = \quad \bigotimes_{\mathrm{J} \in \partial i \setminus \mathrm{I}} \mathrm{q}_{\mathrm{J} \to i}(x_i) \tag{2.2}$$

$$\mathrm{q}_{\mathrm{I} \to i}(x_i) \quad = \quad \bigoplus_{\underline{x}_{\setminus i}} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \bigotimes_{j \in \partial \mathrm{I} \setminus i} \mathrm{q}_{j \to \mathrm{I}}(x_j) \tag{2.3}$$

where equation (2.2) defines the message from a variable to a factor, closer to the root and similarly equation (2.3) defines the message from factor I to a variable i closer to the root. Here, the distributive law allows moving the $\oplus$ over the domain $\mathcal{X}_{\mathrm{I} \setminus i}$ from outside to inside of equation (2.3) – the same way $\oplus$ moves its place in $(a \otimes b) \oplus (a \otimes c)$ to give $a \otimes (b \oplus c)$, where a is analogous to a message.

By starting from the leaves, and calculating the messages towards the root, we obtain the

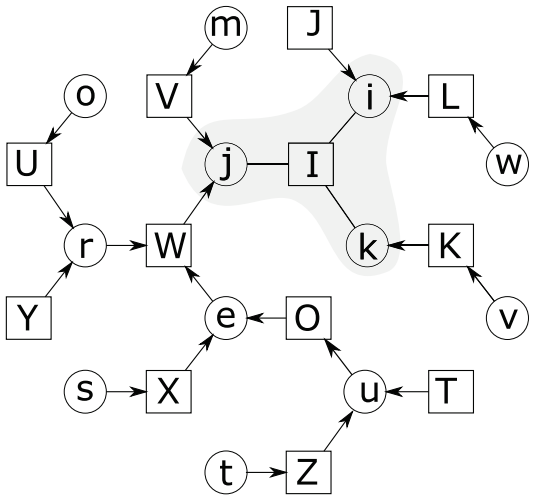


Figure 2.1: *The figure shows a loop-free factor-graph and the direction of messages sent between variable and factor nodes in order to calculate the marginal over the grey region.*

marginal over the root node as the product of incoming messages

$$\mathrm{q}(x_k) \quad = \quad \bigotimes_{\mathrm{I}\in\partial k} \mathrm{q}_{\mathrm{I}\to k}(x_k) \tag{2.4}$$

In fact, we can assume any subset of variables $\underline{x}_{\mathrm{A}}$ (and factors within those variables) to be the root. Then, the set of all incoming messages to A, produces the marginal

$$\mathrm{q}(\underline{x}_{\mathrm{A}}) \quad = \quad \left(\bigotimes_{\mathrm{I}\subseteq\mathrm{A}} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}})\right)\left(\bigotimes_{i\in\mathrm{A},\mathrm{J}\in\partial i,\mathrm{J}\notin\mathrm{A}} \mathrm{q}_{\mathrm{J}\to i}(\underline{x}_i)\right) \tag{2.5}$$

Example 2.1.1. Consider the joint form represented by the factor-graph of figure 2.1

$$\mathrm{q}(\underline{x}) \quad = \quad \bigotimes_{\mathrm{A}\in\{\mathrm{I,J,K,L,O,T,U,V,W,X,Y,Z}\}} \mathrm{f}_{\mathrm{A}}(\underline{x}_{\mathrm{A}})$$

and the problem of calculating the marginal over $\underline{x}_{\{i,j,k\}}$ (*i.e.*, the shaded region).

$$\mathrm{q}(\underline{x}_{\{i,j,k\}}) \quad = \quad \bigoplus_{\underline{x}_{\setminus\{i,j,k\}}} \bigotimes_{\mathrm{A}\in\{\mathrm{I,J,K,L,O,T,U,V,W,X,Y,Z}\}} \mathrm{f}_{\mathrm{A}}(\underline{x}_{\mathrm{A}})$$

We can move the $\oplus$ inside the $\otimes$ to obtain

$$\mathrm{q}(\underline{x}_{\{i,j,k\}}) \quad = \quad \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \otimes \mathrm{q}_{\mathrm{L}\to i}(x_i) \otimes \mathrm{q}_{\mathrm{K}\to i}(x_i) \otimes \mathrm{q}_{\mathrm{V}\to j}(x_j) \otimes \mathrm{q}_{\mathrm{W}\to j}(x_j) \otimes \mathrm{q}_{\mathrm{K}\to k}(x_k)$$

where each term $q_{A\to i}$ factors the summation on the corresponding sub-tree. For example

$$q_{L\to i} \quad = \quad \bigoplus_{x_w} f_L(\underline{x}_L)$$

Here the message $q_{W\to j}$ is itself a computational challenge

$$q_{W\to j} \quad = \quad \bigoplus_{\underline{x}_{\setminus j}} \bigotimes_{A\in\{W,U,Y,X,O,T,Z\}} f_A(\underline{x}_A)$$

However we can also decompose this message over sub-trees

$$q_{W\to j} \quad = \quad \bigoplus_{\underline{x}_{\setminus j}} f_A(\underline{x}_A) \otimes q_{e\to W}(x_e) \otimes q_{r\to W}(x_r)$$

where again using distributive law $q_{e\to W}$ and $q_{r\to W}$ further simplify based on the incoming messages to the variable nodes x_r and x_e.

This procedure is known as Belief Propagation (BP), which is sometimes prefixed with the corresponding semiring *e.g.*, sum-product BP. Even though BP is only guaranteed to produce correct answers when the factor-graph is a tree (and few other cases [8, 22, 310, 313]), it performs surprisingly well when applied as a fixed point iteration to graphs with loops [106, 225]. In the case of loopy graphs the message updates are repeatedly applied in the hope of convergence. This is in contrast with BP on trees, where the messages – from leaves to the root – are calculated only once. The message update can be applied to update the messages either synchronously or asynchronously and the update schedule can play an important role in convergence (*e.g.*, [94, 173]). Here, for numerical stability, when the $\otimes$ operator has an inverse, the messages are normalized. We use $\propto$ to indicate this normalization according to the mode of inference

$$\widehat{p}_{I\to i}(x_i) \quad \propto \quad \bigoplus_{\underline{x}_{\setminus i}} f_I(\underline{x}_I) \bigotimes_{j\in\partial I\setminus i} \widehat{p}_{j\to I}(x_j) \quad \propto \; P_{I\to i}(\widehat{\underline{p}}_{\underline{\partial I\setminus i\to I}})(x_i) \tag{2.6}$$

$$\widehat{p}_{i\to I}(x_i) \quad \propto \quad \bigotimes_{J\in\partial i\setminus I} \widehat{p}_{J\to i}(x_i) \quad \propto \; P_{i\to I}(\widehat{\underline{p}}_{\underline{\partial i\setminus I\to i}})(x_i) \tag{2.7}$$

$$\widehat{p}(\underline{x}_I) \quad \propto \quad f_I(\underline{x}_I) \bigotimes_{i\in\partial I} \widehat{\underline{p}}iI(x_i) \tag{2.8}$$

$$\widehat{p}(x_i) \quad \propto \quad \bigotimes_{I\in\partial i} \widehat{p}_{I\to i}(x_i) \tag{2.9}$$

Here, for general graphs, $\widehat{p}(x_i)$ and $\widehat{p}(\underline{x}_I)$ are approximations to $p(x_i)$ and $p(\underline{x}_I)$ of equation (1.17). The **functionals** $P_{i\to I}(\widehat{p}_{\partial i\setminus I\to i})(.)$ and $P_{I\to i}(\widehat{\underline{p}}_{\underline{\partial I\setminus i\to I}})(.)$ cast the BP message updates as an operator on a subset of incoming messages – *i.e.*, $\widehat{\underline{p}}_{\underline{\partial i\setminus I\to i}} = \{\widehat{p}_{J\to i} \mid J \in \partial i \setminus I\}$. We use these functional notation in presenting the algebraic form of survey propagation in section 2.5.

Another heuristic that is often employed with sum-product and min-sum BP is the **Damping** of messages. This often improves the convergence when BP is applied to loopy graphs. Here a damping parameter $\lambda \in (0,1]$ is used to partially update the new message based on the old one – *e.g.*, for sum-produt BP we have

$$\widehat{\mathrm{p}}_{\mathrm{I}\to i}(x_i) \quad \propto \quad \lambda\widehat{\mathrm{p}}_{\mathrm{I}\to i}(x_i) + (1-\lambda)\Bigg(\sum_{\underline{x}_{\setminus i}} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \prod_{j\in\partial \mathrm{I}\setminus i} \widehat{\mathrm{p}}_{j\to \mathrm{I}}(x_j)\Bigg) \tag{2.10}$$

$$\widehat{\mathrm{p}}_{i\to\mathrm{I}}(x_i) \quad \propto \quad \lambda\widehat{\mathrm{p}}_{i\to\mathrm{I}}(x_i) + (1-\lambda)\Bigg(\prod_{\mathrm{J}\in\partial i\setminus \mathrm{I}} \widehat{\mathrm{p}}_{\mathrm{J}\to i}(x_i)\Bigg) \tag{2.11}$$

$$\tag{2.12}$$

where as an alternative one may use the more expensive form of geometric damping (where λ appears in the power) or apply damping to either variable-to-factor or factor-to-variable messages but not both. Currently – similar to several other ideas that we explore in this thesis – damping is a "heuristic", which has proved its utility in applications but lacks theoretical justification.

2.1.1 Computational Complexity

The time complexity of a single variable-to-factor message update (equation (2.2)) is $O(|\partial i|\ |\mathcal{X}_i|)$. To save on computation, when a variable has a large number of neighbouring factors, and if none of the message values is equal to the annihilator $\overset{\oplus}{1}$ (*e.g.*, zero for the sum-product), and the inverse of $\otimes$ is defined, we can derive the marginals once, and produce variable-to-factor messages as

$$\widehat{\mathrm{p}}_{i\to\mathrm{I}}(x_i) \quad = \quad \widehat{\mathrm{p}}(x_i) \otimes \big(\widehat{\mathrm{p}}_{\mathrm{I}\to i}(x_i)\big)^{-1} \quad \forall \mathrm{I} \in \partial i \tag{2.13}$$

This reduces the cost of calculating all variable-to-factor messages leaving a variable from $O(|\mathcal{X}_i|\ |\partial i|^2)$ to $O(|\mathcal{X}_i|\ |\partial i|)$. We call this type of BP update, **variable-synchronized (v-sync)** update. Note that since max is not Abelian on any non-trivial ordered set, min-max BP does not allow this type of variable-synchronous update. This further motivates using the sum-product reduction of min-max inference. The time complexity of a single factor-to-variable message update (equation (2.3)) is $O(|\mathcal{X}_{\mathrm{I}}|)$. However as we see in section 2.2, sparse factors allow much faster updates. Moreover in some cases, we can reduce the time-complexity by calculating all the factor-to-variable messages that leave a particular factor at the same time (*e.g.*, section 5.2). We call this type of synchronized update, **factor-synchronized (f-sync)** update.

2.1.2 The limits of message passing

By observing the application of distributive law in semirings, a natural question to ask is: can we use distributive law for polynomial time inference on loop-free graphical models over any of the

inference problems at higher levels of inference hierarchy or in general any inference problem with more than one marginalization operation? The answer to this question is further motivated by the fact that, when loops exists, the same scheme may become a powerful approximation technique. When we have more than one marginalization operations, a natural assumption in using distributive law is that the expansion operation distributes over all the marginalization operations – *e.g.*, as in min-max-sum (where sum distributes over both min and max), min-max-min, xor-or-and. Consider the simplest case with three operators $\overset{1}{\oplus}$, $\overset{2}{\oplus}$ and $\otimes$, where $\otimes$ distributes over both $\overset{1}{\oplus}$ and $\overset{2}{\oplus}$. Here the integration problem is

$$q(\emptyset) \quad = \quad \bigoplus^{2}_{\underline{x}_{J_2}} \bigoplus^{1}_{\underline{x}_{J_1}} \bigotimes_{I} f_I(\underline{x}_I)$$

where J_1 and J_2 partition $\{1,\ldots,N\}$.

In order to apply distributive law for each pair $(\overset{1}{\oplus},\otimes)$ and $(\overset{2}{\oplus},\otimes)$, we need to be able to commute $\overset{1}{\oplus}$ and $\overset{2}{\oplus}$ operations. That is, we require

$$\bigoplus^{1}_{\underline{x}_A} \bigoplus^{2}_{\underline{x}_B} g(\underline{x}_{A\cup B}) = \bigoplus^{2}_{\underline{x}_B} \bigoplus^{1}_{\underline{x}_A} g(\underline{x}_{A\cup B}). \tag{2.14}$$

for the specified $A \subseteq J_1$ and $B \subseteq J_2$.

Now, consider a simple case involving two binary variables x_i and x_j, where $g(\underline{x}_{\{i,j\}})$ is

x_i \ x_j	0	1
0	a	b
1	c	d

Applying equation (2.14) to this simple case (*i.e.*, $A = \{i\}, B = \{j\}$), we require

$$(a\overset{1}{\oplus}b)\overset{2}{\oplus}(c\overset{1}{\oplus}d) \quad = \quad (a\overset{2}{\oplus}b)\overset{1}{\oplus}(c\overset{2}{\oplus}d).$$

The following theorem leads immediately to a negative result:

Theorem 2.1.1. *[91]:*

$$(a\overset{1}{\oplus}b)\overset{2}{\oplus}(c\overset{1}{\oplus}d) \;=\; (a\overset{2}{\oplus}b)\overset{1}{\oplus}(c\overset{2}{\oplus}d) \quad \Leftrightarrow \quad \overset{1}{\oplus} = \overset{2}{\oplus} \quad \forall a,b,c$$

which implies that *direct application of distributive law to tractably and exactly solve any inference problem with more than one marginalization operation is unfeasible, even for tree structures.* This limitation was previously known for marginal MAP inference [241].

Min and max operations have an interesting property in this regard. Similar to any other operations for min and max we have

$$\min_{\underline{x}_\mathrm{J}} \max_{\underline{x}_\mathrm{I}} \mathrm{g}(\underline{x}_{\mathrm{I}\cup\mathrm{J}}) \neq \max_{\underline{x}_\mathrm{I}} \min_{\underline{x}_\mathrm{J}} \mathrm{g}(\underline{x}_{\mathrm{I}\cup\mathrm{J}})$$

However, if we slightly change the inference problem (from pure assignments $\underline{x}_{\mathrm{J}_l} \in \mathcal{X}_{\mathrm{J}_l}$ to a distribution over assignments; a.k.a. mixed strategies), as a result of the celebrated *minimax theorem* [300], the min and max operations commute – *i.e.*,

$$\min_{\mathrm{s}(\underline{x}_\mathrm{J})} \max_{\mathrm{s}(\underline{x}_\mathrm{I})} \sum_{\underline{x}_{\mathrm{I}\cup\mathrm{J}}} \mathrm{s}(\underline{x}_\mathrm{J})\mathrm{g}(\underline{x}_{\mathrm{I}\cup\mathrm{J}})\mathrm{s}(\underline{x}_\mathrm{I}) \quad = \quad \max_{\mathrm{s}(\underline{x}_\mathrm{I})} \min_{\mathrm{s}(\underline{x}_\mathrm{J})} \sum_{\underline{x}_{\mathrm{I}\cup\mathrm{J}}} \mathrm{s}(\underline{x}_\mathrm{I})\mathrm{g}(\underline{x}_{\mathrm{I}\cup\mathrm{J}})\mathrm{s}(\underline{x}_{\mathrm{J}_1})$$

where $\mathrm{s}(\underline{x}_{\mathrm{J}_1})$ and $\mathrm{s}(\underline{x}_{\mathrm{J}_2})$ are mixed strategies. This property has enabled addressing problems with min and max marginalization operations using message-passing-like procedures. For example, Ibrahimi et al. [140] solve this (mixed-strategy) variation of min-max-product inference. Message passing procedures that operate on graphical models for game theory (a.k.a. "graphical games") also rely on this property [161, 232].

2.2 Tractable factors

The applicability of graphical models to discrete optimization problems is limited by the size and number of factors in the factor-graph. In section 2.2.1 we review some of the large order factors that allow efficient message passing, focusing on the sparse factors used in part II to solve combinatorial problems. In section 2.2.2 we introduce an augmentation procedure similar to cutting plane method to deal with large number of "constraint" factors.

2.2.1 Sparse factors

The factor-graph formulation of many interesting combinatorial problems involves sparse (high-order) factors. Here, either the factor involves a large number of variables, or the variable domains, $\mathcal{X}_i$, have large cardinality. In all such factors, we are able to significantly reduce the $O(|\mathcal{X}_\mathrm{I}|)$ time complexity of calculating factor-to-variable messages. Efficient message passing over such factors is studied by several works in the context of sum-product and min-sum inference classes [123, 249, 269, 291, 292]. Here we confine our discussion to some of the factors used in part II.

The application of such sparse factors are common in vision. Many image labelling solutions to problems such as image segmentation and stereo reconstruction, operate using priors that enforce similarity of neighbouring pixels. The image processing task is then usually reduced to finding the MAP solution. However pairwise potentials are insufficient for capturing the statistics of natural images and therefore higher-order-factors have been employed [168–170, 174, 183, 234, 268].

The simplest form of sparse factor in combinatorial applications is the **Potts** factor, $\mathrm{f}_{\{i,j\}}(x_i, x_j) = 1(x_i = x_j)$. This factor assumes the same domain for all the variables ($\mathcal{X}_i = \mathcal{X}_j \; \forall i, j$) and its tabular form is non-zero only across the diagonal. It is easy to see that this allows the marginalization of equation (2.3) to be performed in $O(|\mathcal{X}_i|)$ rather than $O(|\mathcal{X}_i| \; |\mathcal{X}_j|)$. Another factor of similar form is the inverse Potts factor, $\mathrm{f}_{\{i,j\}}(x_i, x_j) = 1(x_i \neq x_j)$, which ensures $x_i \neq x_j$. In fact any pair-wise factor that is a constant plus a **band-limited** matrix allows $O(|\mathcal{X}_i|)$ inference (*e.g.*, factors used for bottleneck TSP in section 5.2.2).

Another class of sparse factors is the class of **cardinality factors**, where $\mathcal{X}_i = \{0, 1\}$ and the factor is defined based on only the number of non-zero values –*i.e.*, $\mathrm{f}_\mathrm{I}(\underline{x}_\mathrm{I}) = \mathrm{g}(\sum_{i \in \partial \mathrm{I}} x_i)$. Gail et al. [105] proposes a simple $O(|\partial \mathrm{I}| \; K)$ method for $\mathrm{f}(\underline{x}_\mathrm{I}) = 1((\sum_{i \in \partial \mathrm{I}} x_i) = K)$. We refer to this factor as K-of-N factor and use similar algorithms for at-least-K-of-N $\mathrm{f}_\mathrm{I}(\underline{x}_\mathrm{I}) = 1((\sum_{i \in \partial \mathrm{I}} x_i) \geq K)$ and at-most-K-of-N $\mathrm{f}_\mathrm{I}(\underline{x}_\mathrm{I}) = 1((\sum_{i \in \partial \mathrm{I}} x_i) \leq K)$ factors.

An alternative is the **linear clique potentials** of Potetz and Lee [249]. The authors propose a $O(|\partial \mathrm{I}| \; |\mathcal{X}_i|^2)$ (assuming all variables have the same domain $\mathcal{X}_i$) marginalization scheme for a general family of factors, called linear clique potentials, where $\mathrm{f}_\mathrm{I}(x_\mathrm{I}) = \mathrm{g}(\sum_{i \in \partial \mathrm{I}} x_i w_i)$ for a non-linear $\mathrm{g}_\mathrm{I}(.)$. For sparse factors with larger non-zero values (*i.e.*, larger k), more efficient methods evaluate the sum of pairs of variables using auxiliary variables forming a binary tree and use the Fast Fourier Transform to reduce the complexity of K-of-N factors to $O(|\partial \mathrm{I}| \log(|\partial \mathrm{I}|)^2)$ (see [292] and references in there).

Here for completeness we provide a brief description of efficient message passing through at-least-K-of-N factors for sum-product and min-sum inference.

K of N factors for sum-product

Since variables are binary, it is convenient to assume all variable-to-factor messages are normalized such that $\widehat{\mathrm{p}}_{j \to \mathrm{I}}(0) = 1$. Now we calculate $\widehat{\mathrm{p}}_{\mathrm{I} \to i}(0)$ and $\widehat{\mathrm{p}}_{\mathrm{I} \to i}(1)$ for at-least-K-of-N factors, and then normalize them such that $\widehat{\mathrm{p}}_{\mathrm{I} \to i}(0) = 1$.

In deriving $\widehat{\mathrm{p}}_{\mathrm{I} \to i}(0)$, we should assume that at least K other variables that are adjacent to the factor f_I are nonzero and extensively use the assumption that $\widehat{\mathrm{p}}_{j \to I}(0) = 1$. The factor-to-variable message of equation (2.6) becomes

$$\begin{aligned} \widehat{\mathrm{p}}_{\mathrm{I} \to i}(0) &= \sum_{\underline{x}_{\setminus i}} 1\Big((\sum_{j \in \partial \mathrm{I} \setminus i} x_j) \geq K \Big) \prod_{j \in \partial \mathrm{I} \setminus i} \widehat{\mathrm{p}}_{j \to \mathrm{I}}(x_j) \\ &= \sum_{\mathrm{A} \subseteq \partial \mathrm{I} \setminus i, \; |\mathrm{A}| \geq K} \; \prod_{j \in \mathrm{A}} \widehat{\mathrm{p}}_{j \to \mathrm{I}}(1) \end{aligned} \tag{2.15}$$

where the summation is over all subsets A of $\partial \mathrm{I} \setminus i$ that have at least K members.

Then, to calculate $\widehat{\mathrm{p}}_{\mathrm{I} \to i}(1)$ we follow the same procedure, except that here the factor is replaced

by $1\Big((\sum_{j\in\partial \mathrm{I}\setminus i} x_j) \geq K-1\Big)$. This is because here we assume $x_i = 1$ and therefore it is sufficient for $K-1$ other variables to be nonzero.

Note that in equation (2.15), the sum iterates over "all" $\mathrm{A} \subseteq \partial \mathrm{I} \setminus i$ of size at least K. For high-order factors $\mathrm{f_I}$ (where $|\mathrm{I}|$ is large), this summation contains an exponential number of terms. Fortunately, we can use dynamic programming to perform this update in $O(|\partial \mathrm{I}|\ K)$. The basis for the recursion of dynamic programming is that, starting from $\mathrm{B} = \mathrm{I} \setminus i$, a variable $x_k \in \underline{x}_{\mathrm{K}}$ can be either zero or one

$$\sum_{\mathrm{A}\in\{\mathrm{K}\subseteq\mathrm{B},|\mathrm{K}|\geq k\}} \prod_{j\in\mathrm{A}} \widehat{\mathrm{p}}_{j\to\mathrm{I}}(1) =$$

$$\sum_{\mathrm{A}\in\{\mathrm{K}\subseteq\mathrm{B}\setminus k,\ |\mathrm{K}|\geq K\}} \prod_{j\in\mathrm{A}} \widehat{\mathrm{p}}_{j\to\mathrm{I}}(1) \ + \ \widehat{\mathrm{p}}_{k\to\mathrm{I}}(1)\left(\sum_{\mathrm{A}\in\{\mathrm{K}\subseteq\mathrm{B}\setminus k,\ |\mathrm{A}|\geq K-1\}} \prod_{j\in\mathrm{A}} \widehat{\mathrm{p}}_{j\to\mathrm{I}}(1)\right)$$

where each summation on the r.h.s. can be further decomposed using similar recursion. Here, dynamic program reuses these terms so that each is calculated only once.

K of N factors for min-sum

Here again, it is more convenient to work with normalized variable-to-factor messages such that $\widehat{\mathrm{p}}_{j\to\mathrm{I}}(0) = \overset{\otimes}{1} = 0$. Moreover in computing the factor-to-variable message $\widehat{\mathrm{p}}_{\mathrm{I}\to i}(x_i)$ we also normalize it such that $\widehat{\mathrm{p}}_{\mathrm{I}\to i}(0) = 0$. Recall the objective is to calculate

$$\widehat{\mathrm{p}}_{\mathrm{I}\to i}(x_i) \quad = \quad \min_{\underline{x}_{\setminus i}} 1\Big((\sum_{j\in\partial \mathrm{I}} x_j) = K\Big) \sum_{j\in\partial \mathrm{I}\setminus i} \widehat{\mathrm{p}}_{j\to\mathrm{I}}(x_j)$$

for $x_i = 0$ and $x_i = 1$.

We can assume the constraint factor is satisfied, since if it is violated, the identity function evaluates to $+\infty$ (see equation (1.20)). For the first case, where $x_i = 0$, K out of $|\partial \mathrm{I}\setminus i|$ neighbouring variables to factor I should be non-zero (because $1((\sum_{j\in\partial \mathrm{I}} x_j) = K)$ and $x_i = 0$). The minimum is obtained if we assume the neighbouring variables with smallest $\widehat{\mathrm{p}}_{j\to\mathrm{I}}(1)$ are non-zero and the rest are zero. For $x_i = 1$, only $K-1$ of the remaining neighbouring variables need to be non-zero and therefore we need to find $K-1$ smallest of incoming messages $(\widehat{\mathrm{p}}_{j\to\mathrm{I}}(1)\ \forall j \in \partial \mathrm{I}\setminus i)$ as the rest of messages are zero due to normalization.

By setting the $\widehat{\mathrm{p}}_{\mathrm{I}\to i}(0) = 0$, and letting $\mathrm{A}(K) \subset \partial \mathrm{I}\setminus i$ identify the set of K smallest incoming messages to factor I, the $\widehat{\mathrm{p}}_{\mathrm{I}\to i}(1)$ is given by

$$\widehat{\mathrm{p}}_{\mathrm{I}\to i}(1) \quad = \quad \Big(\sum_{j\in\mathrm{A}(K)} \widehat{\mathrm{p}}_{j\to\mathrm{I}}(1)\Big) - \Big(\sum_{j\in\mathrm{A}(K-1)} \widehat{\mathrm{p}}_{j\to\mathrm{I}}(1)\Big) \quad = \quad \widehat{\mathrm{p}}_{j^K\to\mathrm{I}}(1)$$

where j^K is the index of K^{th} smallest incoming message to I, excluding $\widehat{\mathrm{p}}_{i\to \mathrm{I}}(1)$. A similar procedure can give us the at-least-K-of-N and at-most-K-of-N factor-to-variable updates.

If K is small (*i.e.*, a constant) we can obtain the K^{th} smallest incoming message in $O(K\ |\partial \mathrm{I}|)$ time, and if K is in the order of $|\partial \mathrm{I}|$ this requires $O(|\partial \mathrm{I}| \log(|\partial \mathrm{I}|))$ computations. For both min-sum and sum-product, we incur negligible additional cost by calculating "all" the outgoing messages from factor I simultaneously (*i.e.*, f-sync update).

2.2.2 Large number of constraint factors

We consider a scenario where an (exponentially) large number of factors represent hard constraints (see definition 1.3.2) and ask whether it is possible to find a feasible solution by considering only a small fraction of these constraints. The idea is to start from a graphical model corresponding to a computationally tractable subset of constraints, and after obtaining a solution for a sub-set of constraints (*e.g.*, using min-sum BP), augment the model with the set of constraints that are violated in the current solution. This process is repeated in the hope that we might arrive at a solution that does not violate any of the constraints, before augmenting the model with "all" the constraints. Although this is not theoretically guaranteed to work, experimental results suggest this can be very efficient in practice.

This general idea has been extensively studied under the term **cutting plane methods** in different settings. Dantzig et al. [77] first investigated this idea in the context of TSP and Gomory et al. [118] provided a elegant method to identify violated constraints in the context of finding integral solutions to linear programs (LP). It has since been used to also solve a variety of nonlinear optimization problems. In the context of graphical models, Sontag and Jaakkola [284] (also [286]) use cutting plane method to iteratively tighten the marginal polytope – that enforces the local consistency of marginals; see section 2.3 – in order to improve the variational approximation. Here, we are interested in the augmentation process that changes the factor-graph (*i.e.*, the inference problem) rather than improving the approximation of inference.

The requirements of the cutting plane method are availability of an optimal solver – often an LP solver – and a procedure to identify the violated constraints. Moreover, they operate in real domain $\mathbb{R}^d$; hence the term "plane". However, message passing can be much faster than LP in finding approximate MAP assignments for structured optimization problems [325]. This further motivates using augmentation in the context of message passing.

In sections 4.5 and 5.2, we use this procedure to approximately solve TSP and graph-partitioning respectively. Despite losing the guarantees that make cutting plane method very powerful, augmentative message passing has several advantages: First, message passing is highly parallelizable. Moreover by directly obtaining integral solutions, it is much easier to find violated constraints. (Note the cutting plane method for combinatorial problems operates on *fractional* solutions, whose rounding may eliminate its guarantees.) However, due to non-integral assignments, cutting plane

methods require sophisticated tricks to find violations. For example, see [12] for application of cutting plane to TSP.

2.3 Inference as optimization

The variational approach is concerned with probabilities, and therefore this section is limited to operations on real domain. In the variational approach, **sum-product** inference is expressed as

$$\widehat{\mathsf{p}} \quad = \quad \arg_{\widehat{\mathsf{p}}} \min \quad \mathsf{D}(\widehat{\mathsf{p}} \mid \mathsf{p}^{\beta}) \tag{2.16}$$

where $\mathsf{D}(\widehat{\mathsf{p}} \mid \mathsf{p}^{\beta})$ is the KL-divergence between our approximation $\widehat{\mathsf{p}}$ and the true distribution p at inverse temperature β (see example 1.1.4). Here $\widehat{\mathsf{p}}$ is formulated in terms of desired marginals.

Expanding the definition of KL-divergence and substituting p from equation (1.18), equation (2.16) becomes

$$\widehat{\mathsf{p}} \quad = \quad \arg_{\widehat{\mathsf{p}}} \min \quad \sum_{\underline{x}} \widehat{\mathsf{p}}(\underline{x}) \log(\widehat{\mathsf{p}}(\underline{x})) \; - \; \beta \sum_{\underline{x}} \widehat{\mathsf{p}}(\underline{x}) \log(\mathsf{p}(\underline{x})) \equiv \tag{2.17}$$

$$\arg_{\widehat{\mathsf{p}}} \min \quad \sum_{\underline{x}} \widehat{\mathsf{p}}(\underline{x}) \log(\widehat{\mathsf{p}}(\underline{x})) \; - \; \beta \sum_{\underline{x}} \widehat{\mathsf{p}}(\underline{x}) \left(\sum_{\mathrm{I}} \log(\mathsf{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}})) \right) \tag{2.18}$$

where we have removed the log partition function $\log(\mathsf{q}(\emptyset, \beta)) = \log\left(\sum_{\underline{x}} \prod_{\mathrm{I}} \mathsf{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}})^{\beta}\right)$ from equation (2.17) because it does not depend on $\widehat{\mathsf{p}}$. This means that the minimum of equation (2.18) is $\log(\mathsf{q}(\emptyset, \beta))$, which appears when $\mathsf{D}(\widehat{\mathsf{p}} \mid \mathsf{p}^{\beta}) = 0$ – *i.e.*, $\widehat{\mathsf{p}} = \mathsf{p}^{\beta}$.

The quantity being minimized in equation (2.18), known as **variational free energy**, has two terms: the **(expected) energy** term

$$\mathsf{U}(\widehat{\mathsf{p}}, \mathsf{p}) \stackrel{\text{def}}{=} - \sum_{\underline{x}} \widehat{\mathsf{p}}(\underline{x}) \left(\sum_{\mathrm{I}} \log(\mathsf{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}})) \right)$$

and the **entropy** term

$$\mathsf{H}(\widehat{\mathsf{p}}) = - \sum_{\underline{x}} \widehat{\mathsf{p}}(\underline{x}) \log(\widehat{\mathsf{p}}(\underline{x})) \quad .$$

Different families of representations for $\widehat{\mathsf{p}}(.)$ (in terms of its marginals) produces different inference procedures such as BP, Generalized BP and Mean-field method [303].

Max-product (or **min-sum**) inference is retrieved as zero-temperature limit of sum-product

inference:

$$\begin{aligned}\widehat{\mathrm{p}}(.) &= \lim_{\beta\to+\infty} \arg_{\widehat{\mathrm{p}}} \min \beta \mathrm{U}(\widehat{\mathrm{p}}, \mathrm{p}) - \mathrm{H}(\widehat{\mathrm{p}}) \\ &\equiv \arg_{\widehat{\mathrm{p}}} \min - \sum_{\underline{x}} \widehat{\mathrm{p}}(\underline{x}) \Big(\sum_{\mathrm{I}} \log(\mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}})) \Big) \end{aligned} \tag{2.19}$$

where the energy term is linear in $\widehat{\mathrm{p}}$ and therefore the optima will be at a corner of probability simplex, reproducing the MAP solution.

Here by defining $\mathrm{f}'(\underline{x}) \leftarrow \frac{1}{\mathrm{f}(\underline{x})}$, we get the min-sum form

$$\widehat{\mathrm{p}}(.) = \arg_{\widehat{\mathrm{p}}} \min \sum_{\underline{x}} \widehat{\mathrm{p}}(x_{\mathrm{I}}) \Big(\sum_{\mathrm{I}} \log(\mathrm{f}'_{\mathrm{I}}(\underline{x}_{\mathrm{I}})) \Big)$$

We observe that using a second parameter α, **min-max** inference is also retrievable[1]

$$\begin{aligned}\widehat{\mathrm{p}}(.) &= \lim_{\alpha\to+\infty} \arg_{\widehat{\mathrm{p}}} \min \sum_{\underline{x}} \widehat{\mathrm{p}}(\underline{x}) \Big(\sum_{\mathrm{I}} \log(\mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}))^{\alpha} \Big) \\ &\equiv \arg_{\widehat{\mathrm{p}}} \min \sum_{\underline{x}} \widehat{\mathrm{p}}(\underline{x}_{\mathrm{I}}) \Big(\max_{\mathrm{I}} \log(\mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}})) \Big) \end{aligned} \tag{2.20}$$

where again due to the linearity of the objective in $\widehat{\mathrm{p}}$, the optima are at the extreme points of the probability simplex.

To retrieve sum-product BP update equations from divergence minimization of equation (2.16), we will **reparameterize** $\widehat{\mathrm{p}}$ using its marginals $\widehat{\mathrm{p}}(\underline{x}_{\mathrm{I}})$ and $\widehat{\mathrm{p}}(x_i)$. Here, we present this reparametrization in a more general form, as it holds for a any commutative semiring where $(\mathcal{Y}^*, \otimes)$ is an abelian group.

Proposition 2.3.1. *If the $\otimes$ operator of the semiring has an inverse and the factor-graph is loop-free, we can write $\mathrm{p}(\underline{x})$ as*

$$\widehat{\mathrm{p}}(\underline{x}) = \frac{\bigotimes_{\mathrm{I}} \widehat{\mathrm{p}}(\underline{x}_{\mathrm{I}})}{\bigotimes_{i} \Big(\widehat{\mathrm{p}}(x_i) \odot (|\partial i| - 1) \Big)} \tag{2.21}$$

where the inverse is w.r.t $\otimes$ and the exponentiation operator is defined as $a \odot b \overset{def}{=} \underbrace{a \otimes \ldots \otimes a}_{b \text{ times}}$.

Proof. For this proof we use the exactness of BP on trees and substitute BP marginals equation (2.9)

[1] Here we assume there are no ties at the min-max solution *i.e.*, $\mathrm{f}_{\mathrm{I}}(\underline{x}^*_{\mathrm{I}}) > \mathrm{f}_{\mathrm{J}}(\underline{x}^*_{\mathrm{J}}) \quad \forall \mathrm{J} \neq \mathrm{I}$.

into equation (2.21):

$$\frac{\bigotimes_{\mathrm{I}}\widehat{\mathrm{p}}(\underline{x}_{\mathrm{I}})}{\bigotimes_i\big(\widehat{\mathrm{p}}(x_i)\odot(|\partial i|-1)\big)} = \frac{\bigotimes_{\mathrm{I}}\mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}})\bigotimes_{i\in\partial\mathrm{I}}\widehat{\mathrm{p}}_{i\to\mathrm{I}}(x_i)}{\bigotimes_i\big(\bigotimes_{\mathrm{I}\in\partial i}\widehat{\mathrm{p}}_{\mathrm{I}\to i}(x_i)\odot(|\partial i|-1)\big)} = \frac{\bigotimes_{\mathrm{I}}\mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}})\bigotimes_{i\in\partial\mathrm{I}}\widehat{\mathrm{p}}_{i\to\mathrm{I}}(x_i)}{\bigotimes_i\big(\bigotimes_{\mathrm{I}\in\partial i}\widehat{\mathrm{p}}_{i\to\mathrm{I}}(x_i)\big)} = \bigotimes_{\mathrm{I}}\mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) = \mathrm{p}(\underline{x})$$

where we substituted the variable-to-factor messages in the denominator with factor-to-variable messages according to equation (2.2) and used the definition of inverse (*i.e.*, $a\otimes a^{-1}=\overset{\otimes}{1}$) to cancel out the denominator. □

Intuitively, the denominator is simply cancelling the double counts – that is since $\widehat{\mathrm{p}}(x_i)$ is counted once for any $\mathrm{I}\in\partial i$ in the nominator, the denominator removes all but one of them.

2.3.1 Sum-product BP and friends

Rewriting equation (2.21) for sum-product ring $\widehat{\mathrm{p}}(\underline{x}) = \frac{\prod_{\mathrm{I}}\widehat{\mathrm{p}}(\underline{x}_{\mathrm{I}})}{\prod_i\widehat{\mathrm{p}}(x_i)^{|\partial i|-1}}$ and replacing $\widehat{\mathrm{p}}$ in the variational energy minimization, we get

$$\widehat{\mathrm{p}} = \arg_{\widehat{\mathrm{p}}}\min \quad \beta\sum_{\mathrm{I}}\widehat{\mathrm{p}}(\underline{x}_{\mathrm{I}})\mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \tag{2.22}$$

$$-\left(\sum_{\mathrm{I}}\sum_{\underline{x}_{\mathrm{I}}}\widehat{\mathrm{p}}(\underline{x}_{\mathrm{I}})\log(\widehat{\mathrm{p}}(\underline{x}_{\mathrm{I}}))\right)-\left(\sum_i(1-\mid\partial i\mid)\sum_{x_i}\widehat{\mathrm{p}}(x_i)\log(\widehat{\mathrm{p}}(x_i))\right) \tag{2.23}$$

$$\text{such that} \quad \sum_{x_{\setminus i}}\widehat{\mathrm{p}}(\underline{x}_{\mathrm{I}}) = \widehat{\mathrm{p}}(x_i) \quad \forall i,\mathrm{I}\in\partial i \tag{2.24}$$

$$\sum_{x_i}\widehat{\mathrm{p}}(x_i) = 1 \tag{2.25}$$

where the energy term equation (2.22) is exact and the quantity that is minimized is known as **Bethe free energy** [30, 328]. The constraints equations (2.24) and (2.25) ensure that marginals are consistent and sum to one. Following the lead of Yedidia et al. [328], Heskes [132] showed that stable fixed points of sum-product BP are local optima of Bethe free energy.

The optimization above approximates the KL-divergence minimization of equation (2.18) in two ways: (I) While the marginal constraint ensure local consistency, for general factor-graphs there is no guarantee that even a joint probability $\widehat{\mathrm{p}}$ with such marginals exists (*i.e.*, local consistency conditions outer-bound **marginal polytope**; the polytope of marginals realizable by a join probability $\mathrm{p}(\underline{x})$). (II) Bethe entropy is not exact for loopy factor-graphs. Using the method of Lagrange multipliers to enforce the local consistency constraints and setting the derivatives of equation (2.23) w.r.t. $\widehat{\mathrm{p}}$ to zero, recovers sum-product BP updates [327, 328]. This optimiza-

tion view of inference has inspired many sum-product inference techniques with convex entropy approximations and convergence guarantees [113, 126, 134, 206, 293, 304, 319, 329].

2.3.2 Min-sum message passing and LP relaxation

LP relaxation of min-sum problem seeks marginals $\widehat{\mathrm{p}}(\underline{x}_{\mathrm{I}})\forall \mathrm{I}$

$$\widehat{\mathrm{p}} \quad = \quad \arg_{\widehat{\mathrm{p}}} \min \quad \sum_{\mathrm{I}} \widehat{\mathrm{p}}(\underline{x}_{\mathrm{I}}) \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \tag{2.26}$$

$$\text{such that} \quad \sum_{x_{\setminus i}} \widehat{\mathrm{p}}(\underline{x}_{\mathrm{I}}) = \widehat{\mathrm{p}}(x_i) \quad \forall i, \mathrm{I} \in \partial i \tag{2.27}$$

$$\sum_{x_i} \widehat{\mathrm{p}}(x_i) = 1$$

If integral (*i.e.*, $\widehat{\mathrm{p}}(x_i) = 1(x_i = x_i^*)$ for some $\underline{x}^* = \{x_1^*, \ldots, x_N^*\}$), this LP solution is guaranteed to be optimal (*i.e.*, identical to equation (2.19)). Taking the zero temperature limit ($\lim \beta \to \infty$) of the Bethe free energy of equations (2.22) and (2.23), for any convex entropy approximation [126, 133, 304, 305], ensures that sum-product message passing solution recovers the Linear Programming (LP) solution [311]. Moreover, replacing the summation with maximization (which again corresponds to temperature limit) in the resulting convex message passing, produces the convex min-sum message passing, which agrees with LP relaxations, under some conditions (*e.g.*, when there are no ties in beliefs). The general interest in recovering LP solutions by message passing is to retain its optimality guarantees while benefiting from the speed and scalability of message passing that stems from exploitation of graphical structure [324]. One may also interpret some of these convex variations as replicating variables and factors while keeping the corresponding messages identical over the **replicates** [271]. After obtaining message updates, the number of replicates are allowed to take rational values (Parisi introduced a similar trick for estimation of the partition function using replica trick [55, 211]).

Another notable variation for approximate MAP inference is max-product-linear-program , which performs block coordinate descend in the space of duals for equation (2.26). MPLP is guaranteed to converge and is often able to recover LP solution [114, 285]. Finally **dual (and primal) decomposition** methods minimize factors separately and combine their estimates in a way that agrees with sub-gradient in each iteration [42, 155, 175, 176].

2.3.3 Min-max and other families

By rephrasing the variational min-max inference of equation (2.20)

$$\widehat{\mathrm{p}} = \arg_{\widehat{\mathrm{p}}} \min \sum_{\underline{x}} \widehat{\mathrm{p}}(\underline{x}_{\mathrm{I}}) \Big(\max_{\mathrm{I}} \log(\mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}})) \Big)$$

in terms of marginals $\widehat{\mathrm{p}}(\underline{x}_{\mathrm{I}})$ and enforcing marginal consistency constraints, we obtain the following LP relaxation

$$\begin{aligned} \widehat{\mathrm{p}} \quad &= \quad \arg_{\widehat{\mathrm{p}}} \min \quad y \\ \text{such that} \quad & \sum_{\underline{x}_{\mathrm{I}}} \widehat{\mathrm{p}}(\underline{x}_{\mathrm{I}}) \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \leq y \quad \forall \mathrm{I} \\ & \sum_{x_{\setminus i}} \widehat{\mathrm{p}}(\underline{x}_{\mathrm{I}}) = \widehat{\mathrm{p}}(x_i) \quad \forall i, \mathrm{I} \in \partial i \\ & \sum_{x_i} \widehat{\mathrm{p}}(x_i) = 1 \end{aligned} \tag{2.28}$$

which surprisingly resembles our sum-product reduction of min-max inference in section 1.4.2. Here $\sum_{\underline{x}_{\mathrm{I}}} \widehat{\mathrm{p}}(\underline{x}_{\mathrm{I}}) \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \leq y$ is a relaxation of our sum-product factor $1(\mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \leq y)$ in equation (1.27).

Claim 2.3.2. *y in equation (2.28) lower bounds the min-max objective y^*. Moreover, if $\widehat{\mathrm{p}}$ is integral, then $y = y^*$ and $\underline{x}^*$ is the optimal min-max assignment.*

Proof. The integral solution $\widehat{\mathrm{p}}$, corresponds to the following optimization problem

$$\begin{aligned} \underline{x}^* \quad &= \quad \arg_{\underline{x}} \min \quad y \\ \text{such that} \quad & \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \leq y \\ & \equiv \quad \arg_{\underline{x}} \min \ \max_{\mathrm{I}} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \end{aligned} \tag{2.29}$$

which is the exact min-max inference objective. Therefore, for integral $\widehat{\mathrm{p}}$, we obtain optimal min-max solution. On the other hand by relaxing the integrality constraint, because of the optimality guarantee of LP, the LP solution y can not be worse than the integral solution and its corresponding value y^*. □

This lower bound complements the upper bound that we obtain using a combination of sum-product reduction and an incomplete solver (such as perturbed BP of section 2.6.3) and can be used to assess the optimality of a min-max solution.

The only other extensively studied inference problem in the inference hierarchy of section 1.2 is **max-sum-product** (a.k.a. marginal MAP) inference [80, 152, 202]. In particular variational formulation of max-sum-product inference [195], substitutes the entropy term in equation (2.17) with conditional entropy.

2.4 Loop corrections

In this section we first review the region-based methods that account for short loops in section 2.4.1 and then show how to perform loop correction by taking into account the message dependencies

in section 2.4.2. In section 2.4.3 we introduce a loop correction method that can benefit from both types of loop corrections, producing more accurate marginals. While the region-based techniques can be used to directly estimate the integral, the approximation techniques that take message dependencies into account are only applied for estimation of marginals.

2.4.1 Short loops

We consider a general class of methods that improve inference in a loopy graphical model by performing exact inference over regions that contain small loops.

The earliest of such methods is **junction-tree** [148, 186], which performs exact inference with computation cost that grows exponentially in the size of largest region –*i.e.*, tree width [57]. Here, regions form a tree and the messages are passed over regional intersections. While this algorithm is still popular in applications that involve certain class of graphs [35] or when the exact result is required, most graphs do not have a low tree width [157, 167].

An extension to junction tree is the **junction graph** method [6] that removes the requirement for the regions to form a tree. For this, the proxy between two regions is a subset of their intersection (rather than the whole intersection) and one still requires the regions that contain a particular variable to form a tree. Similar ideas are discussed under the name of cluster graphs in [171].

Inspired by the connection between Bethe free energy and belief propagation (see section 2.3), Yedidia et al. [328] proposed **Generalized BP** that minimizes Kikuchi approximation to free energy (a.k.a. **Cluster Variational Method** [165, 246]). Here the entropy approximation is obtained from a region-graph.

A **region** ρ is a collection of connected variables $\mathcal{V}(\rho)$ and a set of factors $\mathcal{F}(\rho)$ such that each participating factor depends only on the variables included in the region. To build the CVM region-graph[2], one starts with predefined top (or outer) regions such that each factor is included in at least one region. Then, we add the intersection of two regions (including variables and factors) recursively until no more sub(inner)-region can be added. Each region is then connected to its immediate parent.

A region-graph, reparameterizes $\widehat{\mathrm{p}}(.)$ in terms of its marginals over the regions

$$\widehat{\mathrm{p}}(\underline{x}) \quad = \quad \prod_{\rho} \widehat{\mathrm{p}}(\underline{x}_{\mathcal{V}(\rho)})^{\mathrm{c}(\rho)} \tag{2.30}$$

where $\mathrm{c}(\rho)$ is the **counting number** for region ρ and ensures that each variable and factor is

[2]Here we are making a distinction between a general region graph and a CVM region-graph.

counted only once. This number is recursively defined by Möbius formula for inner regions:

$$c(\rho) \quad = \quad 1 - \sum_{\rho' \supset \rho} c(\rho) \tag{2.31}$$

where ρ' is an ancestors of ρ in the region graph.[3]

Similar to BP, by substituting the reparametrization of equation (2.30) into the variational free energy minimization of equation (2.18) we get

$$\begin{aligned} \widehat{p} \;=\; & \arg_{\widehat{p}} \min \sum_{\rho} c(\rho) \left(\sum_{\underline{x}_{\mathcal{V}(\rho)}} \widehat{p}(\underline{x}_{\mathcal{V}(\rho)}) \left(\Big(\sum_{I \in \mathcal{F}(\rho)} f_I(\underline{x}_I) \Big) - \widehat{p}(\underline{x}_{\mathcal{V}(\rho)}) \log(\widehat{p}(\underline{x}_{\mathcal{V}(\rho)})) \right) \right) \\ \text{s.t.} \quad & \sum_{\underline{x}_{\setminus \mathcal{V}(\rho)}} \widehat{p}(\underline{x}_{\mathcal{V}(\rho')}) = \widehat{p}(\underline{x}_{\mathcal{V}(\rho)}) \quad \forall \rho \subset \rho' \end{aligned} \tag{2.32}$$

which is known as **Kikuchi approximation** to free energy [165]. The constraints of equation (2.32) ensure that marginals are consistent across overlapping regions. Solving this constraint optimization using the method of Lagrange multipliers, yields a set of recursive equations that are known as Generalized BP equations [328]. Again a region-based approximation is exact only if the region-graph has no loops.

A region-graph without restrictions on the choice of regions generalizes junction-graph method as well. The general construction of the region graph only requires that the counting numbers of all the regions to which a variable (or a factor) belong, sum to 1 [327]. For different criteria on the choice of regions see also [235, 314, 317].

2.4.2 Long loops

In graphical models with long-range correlation between variables, region-based methods are insufficient. This is because their complexity grows exponentially with the number of variables in each region and therefore they are necessarily inefficient account for long loops in the graph.

A class of methods for reducing long-range correlations are methods based on **cut-set conditioning** [245], where a subset of variables are clamped, so as to remove the long-range correlations that are formed through the paths that include the cut-set. For example, consider a Markov network in the form of a cycle. By fixing any single variable, the reduced factor graph becomes a tree and therefore allows exact inference. Several works investigate more sophisticated ideas in performing better inference by clamping a subset and the resulting theoretical guarantees [78, 89, 312]. A closely related idea is **Rao-Blackwellization** (a.k.a. collapsed MCMC; see section 2.6.1), a hybrid approach to inference [108] where particles represent a partial assignment of variables and inference over the rest of variables is performed using a deterministic method. The deterministic

[3] More accurately $\mathcal{V}(\rho) \subseteq \mathcal{V}(\rho')$.

inference method such as BP is used to calculate the value of the partition function, for each possible joint assignment of the variables that are not collapsed. Then collapsed particles are sampled accordingly. This process in its general form is very expensive, but one could reduce the cost, depending on the structure of the network [32].

The **loop calculus** of Chertkov [60][61] expands the free energy around the Bethe approximation, with one term per each so-called generalized loop in the graph. Since the number of loops grows exponentially in the number of variables, this expansion does not provide a practical solution. Some attempts have been made to to make this method more practical by truncating the loop series [117]. While the original loop series was proposed for binary valued and pairwise factors, it has been generalized to arbitrary factor-graphs [315, 320] and even region-graphs [334].

Another class of approximate inference methods perform loop correction by estimating the message dependencies in a graphical model [217, 220, 262]. These methods are particularly interesting as they directly compensate for the violated assumption of BP – *i.e.*, corresponding to independent set of incoming messages.

For the benefit of clarity, we confine the loop correction equations in this section and its generalization in the next section to Markov networks (*i.e.*, $|\partial \mathrm{I}| = 2$); see [254] for our factor-graph versions. Although the previous works on loop corrections have been only concerned with sum-product inference, here we present loop corrections for a general commutative semiring $(\boldsymbol{\mathcal{Y}}^*, \oplus, \otimes)$ in which the operation $\otimes$ has an inverse (*i.e.*, $(\boldsymbol{\mathcal{Y}}^*, \otimes)$ is a group). In particular this means these loop corrections may be used for min-sum class of inference.

Here we rewrite BP update equations (2.2) and (2.3) for Markov networks[4]

$$\widehat{\mathsf{p}}_{i\to j}(x_i) \quad \propto \quad \bigoplus_{\backslash x_i} \bigotimes_{k\in\Delta i\backslash j} \mathsf{f}_{\{k,i\}}(\underline{x}_{\{k,i\}}) \otimes \widehat{\mathsf{p}}_{k\to i}(x_k) \tag{2.33}$$

$$\widehat{\mathsf{p}}(x_i) \quad \propto \quad \bigoplus_{\backslash x_i} \bigotimes_{k\in\Delta i} \mathsf{f}_{\{k,i\}}(\underline{x}_{k,i})\ \widehat{\mathsf{p}}_{k\to i}(x_k) \tag{2.34}$$

Figure 2.2(left) shows the BP messages on a part of Markov network. Here if the Markov network is a tree, BP's assumption that $\widehat{\mathsf{p}}_{s\to i}$, $\widehat{\mathsf{p}}_{o\to i}$ and $\widehat{\mathsf{p}}_{n\to i}$ are independent is valid, because these messages summarize the effect of separate sub-trees on the node i. However if the graph has loops, then we use $\widehat{\mathsf{h}}(\underline{x}_{\Delta i\backslash j})$ to denote **message dependencies**. If we had access to this function, we could easily change the BP message update of equation (2.33) to

$$\widehat{\mathsf{p}}_{i\to j}(x_i) \quad \propto \quad \widehat{\mathsf{h}}(\underline{x}_{\Delta i\backslash j}) \otimes \bigoplus_{\backslash x_i} \bigotimes_{k\in\Delta i\backslash j} \mathsf{f}_{\{k,i\}}(\underline{x}_{\{k,i\}}) \otimes \widehat{\mathsf{p}}_{k\to i}(x_k)$$

Since it is not clear how to estimate $\mathsf{h}(\underline{x}_{\Delta i\backslash j})$, we follow a different path, and instead estimate

[4] Note that $\widehat{\mathsf{p}}_{i\to j}(x_i)$ is over $\mathcal{X}_i$ rather the conventional way of defining it on $\mathcal{X}_j$. This formulation is the same as original BP equation for Markov network if the graph does not have a loop of size two.

the so-called **cavity distribution**, which we denote by $\widehat{\mathsf{h}}(\underline{x}_{\Delta i})$, which is simply the joint marginal over the Markov blanket after making a cavity – *i.e.*, removing a variable i and its neighbouring factors $\mathsf{f}_I \; \forall I \in \partial i$. However, since the missing information is the "dependence" between the messages, $\widehat{\mathsf{h}}(\underline{x}_{\Delta(i)})$ has a degree of freedom w.r.t. individual marginals – *i.e.*, it can be inaccurate by a factor of $\bigotimes_{j\in\Delta(i)} \mathsf{g}_j(x_j)$ for some $\mathsf{g}_j \forall j$, without affecting the loop-corrected message passing procedure. This degree of freedom is essentially equivalent to the freedom in initialization of BP messages. In the following, we show the resulting loop-corrected message passing. But first we write BP updates in a different form.

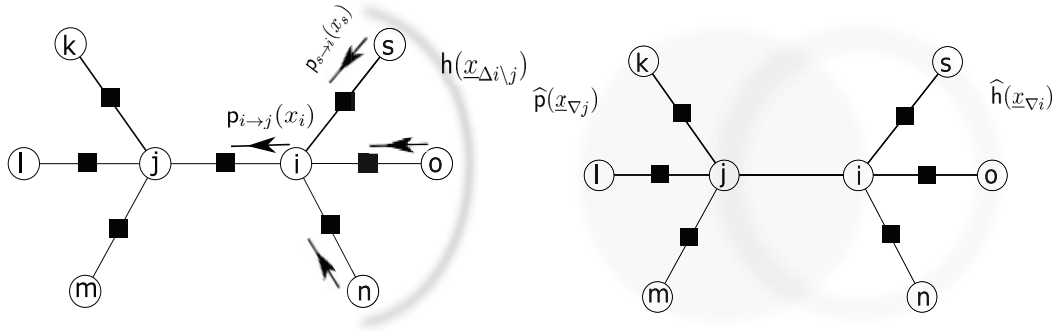


Figure 2.2: *(left) BP messages on a Markov network and the ideal way dependencies should be taken into account. (right) BP marginal over extended Markov blanket ∇j for node j and the message dependencies over the Markov blanket Δi for node i.*

Define the **extended Markov blanket** $\nabla i = \Delta i \cup \{i\}$ be the Markov blanket of i plus i itself, see figure 2.2 (right). We can write BP marginals over ∇i

$$\widehat{\mathsf{p}}(\underline{x}_{\nabla i}) \propto \bigotimes_{k\in\Delta i} \mathsf{f}_{\{i,k\}}(\underline{x}_{\{i,k\}}) \otimes \widehat{\mathsf{p}}_{k\to i}(x_k) \tag{2.35}$$

Using this, equation (2.33) simplifies to:

$$\widehat{\mathsf{p}}_{i\to j}(x_i) \propto \bigoplus_{\setminus x_i} \widehat{\mathsf{p}}(\underline{x}_{\nabla i})/\mathsf{f}_{\{i,j\}}(\underline{x}_{\{i,j\}}) \tag{2.36}$$

$$\widehat{\mathsf{p}}(x_k) \propto \bigoplus_{\setminus x_i} \widehat{\mathsf{p}}(\underline{x}_{\nabla i}) \tag{2.37}$$

Now assume we are given the dependency between the messages $\widehat{\mathsf{p}}_{k\to i}(x_k) \; \forall k \in \Delta i$ in the form of $\widehat{\mathsf{h}}(\underline{x}_{\Delta i})$. This means we can re-express equation (2.35) as

$$\widehat{\mathsf{p}}(\underline{x}_{\nabla i}) \propto \widehat{\mathsf{h}}(\underline{x}_{\Delta i}) \bigotimes_{k\in\Delta i} \mathsf{f}_{\{i,k\}}(\underline{x}_{\{i,k\}}) \otimes \widehat{\mathsf{p}}_{k\to i}(x_k) \tag{2.38}$$

By enforcing the marginal consistency of $\widehat{\mathrm{p}}(\underline{x}_{\nabla i})$ and $\widehat{\mathrm{p}}(\underline{x}_{\nabla j})$ over x_i (and x_j)

$$\bigoplus_{\underline{x}_{\setminus i,j}} \widehat{\mathrm{p}}(\underline{x}_{\nabla i})/\mathrm{f}_{\{i,j\}}(x_i,x_j) \quad = \quad \bigoplus_{\underline{x}_{\setminus i,j}} \widehat{\mathrm{p}}(\underline{x}_{\nabla j})/\mathrm{f}_{\{i,j\}}(x_i,x_j)$$

we retrieve a message update similar to that of BP (in equation (2.33)) that incorporates the dependency between BP messages

$$\widehat{\mathrm{p}}^{(\mathrm{t}+1)}_{i\to j}(x_i) \propto \frac{\bigoplus_{\setminus x_i} \widehat{\mathrm{p}}(\underline{x}_{\nabla_i})/\mathrm{f}_{\{i,j\}}(\underline{x}_{\{i,j\}})}{\bigoplus_{\setminus x_i} \widehat{\mathrm{p}}(\underline{x}_{\nabla_j})/\mathrm{f}_{\{i,j\}}(\underline{x}_{\{i,j\}})} \otimes \widehat{\mathrm{p}}^{(\mathrm{t})}_{i\to j}(x_i) \tag{2.39}$$

It is easy to verify that this update reduces to BP updates (equation (2.33)) when $\widehat{\mathrm{h}}(\underline{x}_{\partial i})$ is uniform – that is we do not have any dependency between messages. The loop-correction method of Mooij *et al.* [220] is similar, however this interpretation does not apply to their updates for factor graphs. We extend the same idea to perform loop-correction for overlapping regions of connected variables in section 2.4.3 where we pass the messages from one region to the outer boundary of another region.

The main computational cost in these loop correction methods is estimating $\widehat{\mathrm{h}}(\underline{x}_{\Delta i})$, the message dependencies. We use clamping to perform this task. For this we remove x_i and all the immediately depending factors from the graph. Then we approximate the marginal $\widehat{\mathrm{h}}(\underline{x}_{\Delta i})$ by reduction to integration; see section 1.4.1. Note that the $\widehat{\mathrm{h}}(\underline{x}_{\Delta i})$ obtained this way contains not only dependencies but also the individual marginals in the absence of node i ($\widehat{\mathrm{h}}(x_j)\ \forall j \in \Delta i$). However since the messages updates for $\widehat{\mathrm{p}}_{j\to i}(x_j)\ \forall j \in \Delta i$, perform "corrections" to this joint probability, we do not need to divide $\widehat{\mathrm{h}}(\underline{x}_{\Delta i})$ by the individual marginals.

2.4.3 Both message dependencies and short loops

Section 2.4.1 presented loop correction methods that improve loopy BP by considering interactions within small clusters of variables, thus taking small loops within these clusters into account. The previous section showed how to account for dependency between BP messages – thus taking long-range correlations into account. In this section we introduce a generalization that performs both types of loop correction.

The basic idea is to form regions, and perform exact inference over regions, to take short loops into account. However in performing message passing between these regions, we introduce a method to perform loop correction over these messages.

We start by defining a **region** $\rho = \{i,\ldots,l\}$ as a set of connected variables. Note that this definition is different from definition of region for region-based methods as it only specifies the set of variables, and not factors. Let $\Delta\rho = \{i \in \Delta j, i \notin \rho \mid j \in \rho\}$ be the Markov blanket of region ρ, and as before let $\nabla\rho = \rho \cup \Delta\rho$.

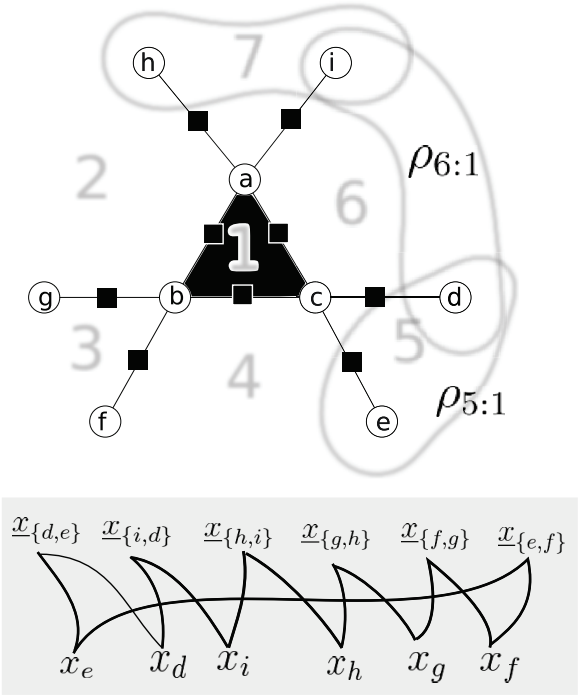


Figure 2.3: *(top) seven regions $\rho_1, \ldots, \rho_7$ and the domain of messages sent from each region to ρ_1. Here $\rho_{5:1}$ is the domain of message from region 5 to region 1. (bottom) the message region-graph shows how these overlapping messages are combined to prevent double-counts.*

Region ρ_1 is a neighbour of ρ_2 with **neighbourhood** $\rho_{1:2}$ *iff* $\rho_{1:2} \stackrel{\text{def}}{=} (\Delta\rho_1) \cap \rho_2 \neq \emptyset$ – *i.e.*, the Markov blanket of ρ_1 intersects with ρ_2 (note that $\rho_{1:2}$ is different from $\rho_{2:1}$). The messages are exchanged on these neighbourhoods and $\widehat{\mathrm{p}}_{1\to 2}(\underline{x}_{\rho_{1:2}})$ is a message from region ρ_1 to ρ_2.

Example 2.4.1. Figure 2.3 shows a set of neighbouring regions (indexed by $1, 2, 3, 4, 5, 6$ and 7). Here the region ρ_1 receives "overlapping" messages from four other regions. For example the message $\widehat{\mathrm{p}}_{6\to 1}(\underline{x}_{d,i})$ overlaps with the message $\widehat{\mathrm{p}}_{5\to 1}(\underline{x}_{d,e})$ as well as $\widehat{\mathrm{p}}_{7\to 1}(\underline{x}_{h,i})$. Therefore simply writing $\mathrm{p}(\underline{x}_{\nabla\rho})$ in terms of the factors inside $\nabla\rho$ and the incoming messages (as in equation (2.5)) will double-count some variables.

Message region-graphs

Here, similar to section 2.4.1, we construct a region-graph to track the double-counts. However, here we have one message-region-graph per region ρ. The construction is similar to that of cluster variational methods; we start with the original message-domains (*e.g.*, $\rho_{2:1}$) and recursively add the intersections, until no more message-region γ can be added. Each message-region γ is connected to its immediate parent. figure 2.3 shows the two-layered message-region-graph for region ρ_1. Here, for discussions around a particular message-region-graph we will drop the region-index 1.

Let $\mathsf{m}(\underline{x}_\gamma) \stackrel{\text{def}}{=} \widehat{\mathsf{p}}_{\pi\to 1}(\underline{x}_{\rho_{\pi:1}})\ \forall\pi$ be the top regions in the region-graph, consisting of all the incoming messages to region of interest ρ_1. The Möbius formula (equation (2.31)) gives counting number for message-region (here again the top regions' counting number is one). A **downward pass**, starting from top regions, calculates the belief $\mathsf{m}(\underline{x}_\gamma)$ over each message-region, as the average of beliefs over its parents.

Example 2.4.2. In figure 2.3 the belief $\mathsf{m}(x_e)$ is the average of beliefs over $\mathsf{m}(\underline{x}_{\{d,e\}}) = \widehat{\mathsf{p}}_{5\to 1}(\underline{x}_{\rho_{5:1}})$ and $\mathsf{m}(\underline{x}_{\{e,h\}}) = \widehat{\mathsf{p}}_{4\to 1}(\underline{x}_{\rho_{4:1}})$ when marginalized over x_e. Here the counting number of x_e is $\mathsf{c}(\{e\}) = 1-(1+1) = -1$.

We require the $\otimes$ operator of the semiring to have an inverse. Recall the power operator $\odot\ y\odot k \stackrel{\text{def}}{=} \underbrace{y\otimes\ldots\otimes y}_{\text{k times}}$. For sum-product and min-sum semirings, this operator corresponds to exponentiation and product respectively and it is well-defined also for rational numbers k. Now define the average as

$$\mathsf{avg}(\{y_1,\ldots,y_k\}) \stackrel{\text{def}}{=} \Big(\bigotimes_i y_i\Big)\odot\frac{1}{k} \tag{2.40}$$

Using $\mathsf{Pa}(\gamma)$ to denote the parents of region γ, in the downward pass

$$\mathsf{m}(\underline{x}_\gamma) \propto \bigotimes_{\gamma'\in\mathsf{Pa}(\gamma)} \mathsf{avg}\Big(\bigoplus_{\underline{x}_{\setminus\gamma}} \mathsf{m}(\gamma')\Big) \tag{2.41}$$

where $\mathsf{m}(\underline{x}_\gamma)$ for top regions are just the incoming messages.

Let $\mathsf{f}_\mathrm{A}(\underline{x}_\mathrm{A}) = \bigotimes_{\mathrm{I}\subseteq\mathrm{A}} \mathsf{f}_\mathrm{I}(\underline{x}_\mathrm{I})$ be the semiring-product of all the factors defined over a subset of A. For example in figure 2.3, $\mathsf{f}_{\nabla\rho_a}(\underline{x}_{\nabla\rho_a})$ is the product of 9 pairwise factors (*i.e.*, all the edges in the figure).

After a downward pass, the belief over $\nabla\rho$ (analogous to equation (2.35)):

$$\widehat{\mathsf{p}}(\underline{x}_{\nabla\rho}) \quad\propto\quad \mathsf{f}_{\nabla\rho}(\underline{x}_{\nabla\rho}) \otimes \Big(\bigotimes_\rho \mathsf{m}(\underline{x}_\gamma)\odot\mathsf{c}(\gamma)\Big) \tag{2.42}$$

that is $\widehat{\mathsf{p}}(\underline{x}_{\nabla\rho})$ is the semiring product of all the factors inside this region and all the beliefs over message-regions inside its message-region-graph, where double counts are taken into account. For our example, assuming sum-product ring,

$$\widehat{\mathsf{p}}(\underline{x}_{\nabla\rho_a}) = \mathsf{f}_{\nabla\rho_a}(\underline{x}_{\nabla\rho_a})\mathsf{m}(\underline{x}_{4,5})\ldots\mathsf{m}(\underline{x}_{9,4})\Big(\mathsf{m}(x_4)^{-1}\ldots\mathsf{m}(x_9)^{-1}\Big)$$

where the semiring-product and inverse are product and division on real domain.

At this point we can also introduce an estimate for message dependencies $\widehat{\mathsf{h}}(\underline{x}_{\partial\rho})$ into the

equation above, and generalize the update of equation (2.39) to

$$\widehat{\mathrm{p}}^{(\mathrm{t}+1)}_{b\to a}(\underline{x}_{\rho_{b:a}}) \propto \frac{\bigoplus_{\backslash \underline{x}_{\rho_{b:a}}} \widehat{\mathrm{p}}(\underline{x}_{\nabla\rho_b})/\mathrm{f}_{\nabla\rho_a\cap\nabla\rho_b}(\underline{x}_{\nabla\rho_a\cap\nabla\rho_b})}{\bigoplus_{\backslash x s_{\rho_{b:a}}} \widehat{\mathrm{p}}(\underline{x}_{\nabla\rho_a})/\mathrm{f}_{\nabla\rho_a\cap\nabla\rho_b}(\underline{x}_{\nabla\rho_a\cap\nabla\rho_b})} \otimes \tilde{\mathrm{p}}^{(\mathrm{t})}_{b\to a}(\underline{x}_{\rho_{b:a}}) \tag{2.43}$$

One last issue to resolve is to define the "effective" message $\tilde{\mathrm{p}}^{(\mathrm{t})}_{b\to a}(\underline{x}_{\rho_{b:a}})$, which is different from $\widehat{\mathrm{p}}^{(\mathrm{t})}_{b\to a}(\underline{x}_{\rho_{b:a}})$. Since we did not directly use $\widehat{\mathrm{p}}^{(\mathrm{t})}_{b\to a}(\underline{x}_{\rho_{b:a}})$ in the previous iteration, we should not include it directly in this update. Instead we use the message region-graph for region a to calculate the effective message:

$$\tilde{\mathrm{p}}_{b\to a}(\underline{x}_{\rho_{b:a}}) = \bigotimes_{\gamma\subseteq\rho_{b:a}} \Big(\mathrm{m}(\underline{x}_\gamma)\odot \mathrm{c}(\gamma)\Big) \tag{2.44}$$

The effective message, as defined above (equation (2.44)), can be efficiently calculated in an **upward pass** in the message region-graph. Starting from the parents of the lowest regions, update the belief $\mathrm{m}(\underline{x}_\gamma)$ obtained in downward pass equation (2.41) using the new beliefs over its children:

$$\tilde{\mathrm{m}}(\underline{x}_\gamma) \quad = \quad \mathrm{m}(\underline{x}_\gamma) \bigotimes_{\gamma'\in \mathrm{Ch}(\gamma)} \frac{\tilde{\mathrm{m}}(\underline{x}_{\gamma'})}{\bigoplus_{\underline{x}_{\backslash\gamma'}} \mathrm{m}(\underline{x}_\gamma)} \tag{2.45}$$

where $\mathrm{Ch}(\gamma)$ is the set of children of γ in the message region-graph. After the upward pass, the new beliefs over top regions gives us the effective messages

$$\tilde{\mathrm{p}}_{a\to b}(\underline{x}_{\rho_{a:b}}) = \tilde{\mathrm{m}}(\underline{x}_{\rho_{a:b}}) \tag{2.46}$$

Example 2.4.3. In our example of figure 2.3(bottom), assuming a sum-product ring, since the message-region-graph only has two layers, we can write the effective message $\tilde{\mathrm{p}}_{h\to a}(\underline{x}_{\{4,9\}})$ as

$$\tilde{\mathrm{p}}_{h\to a}(\underline{x}_{\{4,9\}}) = \widehat{\mathrm{p}}_{h\to a}(\underline{x}_{\{4,9\}}) \frac{\mathrm{m}(x_4)\mathrm{m}(x_9)}{(\sum_{x_4} \widehat{\mathrm{p}}_{h\to a}(\underline{x}_{\{4,9\}}))(\sum_{x_9} \widehat{\mathrm{p}}_{h\to a}(\underline{x}_{\{4,9\}}))} \tag{2.47}$$

This form of loop correction, which we call Generalized Loop Correction (GLC), generalizes both correction schemes of sections 2.4.1 and 2.4.2. The following theorem makes this relation to generalized BP more explicit.

Theorem 2.4.1. [5] *For Markov networks, if the regions $\{\rho\}$ partition the variables, then any Generalized BP fixed point of a particular region-graph construction is also a fixed point for GLC, when using uniform message dependencies* (i.e., $\widehat{\mathrm{h}}(\underline{x}_{\Delta i}) \propto 1\ \forall i, \underline{x}_{\Delta i}$).

[5] See [254] for the proof.

2.4.4 Experiments

This section compares different variations of our generalized loop correction method (GLC) for sum-product ring against BP as well as Cluster Variational Method (CVM; section 2.4.1), loop correction of [220] (LCBP), which does not exactly account for short loops and the Tree Expectation Propagation (TreeEP) [214] method, which also performs some kind of loop correction. For CVM, we use the double-loop algorithm [133], which is slower than Generalized BP but has better convergence properties.[6] We report the time in seconds and the error for each method as the average of absolute error in single variable marginals – *i.e.*, $\frac{1}{N}\sum_i \sum_{\setminus x_i} \mid \widehat{\mathsf{p}}(x_i) - \mathsf{p}(x_i) \mid$. For each setting, we report the average results over 10 random instances of the problem. We experimented with grids and 3-regular random graphs.[7]

Both LCBP and GLC can be used without any information on message dependencies. with an initial cavity distribution estimated via clamping cavity variables. In the experiments, *full* means message dependencies $\widehat{\mathsf{h}}$ was estimated while *uniform* means $\widehat{\mathsf{h}}$ was set to uniform distribution (*i.e.*, loop-correction was over the regions only). We use GLC to denote the case where the regions were selected such that they have no overlap (*i.e.*, $\rho_a \cap \rho_b = \emptyset \; \forall a, b$) and GLC+ when overlapping clusters of some form are used. For example, *GLC+(Loop4, full)* refers to a setting with message dependencies that contains all overlapping loop clusters of length up to 4. If a factor does not appear in any loops, it forms its own cluster. The same form of clusters are used for CVM.

Grids

We experimented with periodic spin-glass Ising grids of example 1.1.4 on page 12. In general, smaller local fields and larger variable interactions result in more difficult problems. We sampled local fields independently from $\mathcal{N}(0,1)$ and interactions from $\mathcal{N}(0,\beta^2)$. Figure 2.4a summarizes the results for 6x6 grids for different values of β.

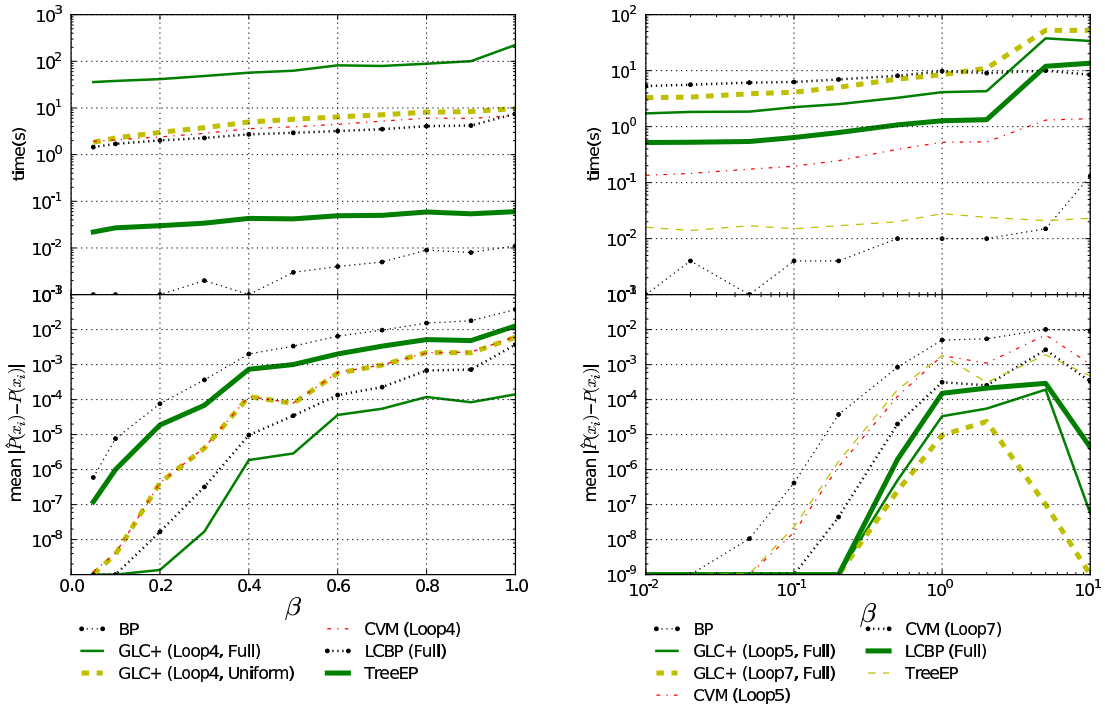
We also experimented with periodic grids of different sizes, generated by sampling all factor entries independently from $\mathcal{N}(0,1)$. Figure 2.5a compares the computation time and error of different methods for grids of sizes that range from 4x4 to 10x10.

Regular Graphs

We generated two sets of experiments with random 3-regular graphs (all nodes have degree 3) over 40 variables. Here we used Ising model when both local fields and couplings are independently sampled from $\mathcal{N}(0,\beta^2)$. Figure 2.4b shows the time and error for different values of β. Figure 2.5b shows time versus error for graph size between 10 to 100 nodes for $\beta = 1$.

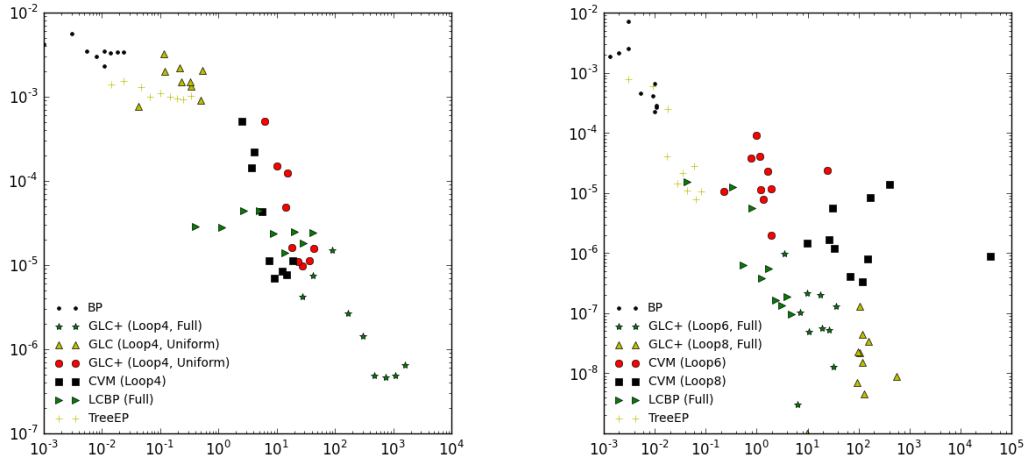
[6] All methods are applied without any damping. We stop each method after a maximum of 1E4 iterations or if the change in the probability distribution (or messages) is less than 1E-9.

[7] The evaluations are based on implementation in *libdai* inference toolbox [221].



(a) *spin-glass Ising grid* ***(b)*** *spin-glass Ising model on a 3-regular graph*

Figure 2.4: *Average Run-time and accuracy for 6x6 spinglass Ising grids and 3-regular Ising model for different values of β. Variable interactions are sampled from $\mathcal{N}(0,\beta^2)$ and local fields are sampled from $\mathcal{N}(0,1)$.*



(a) *spin-glass Ising grid* ***(b)*** *spin-glass Ising model on 3-regular graph*

Figure 2.5: *Time vs error for Ising grids and 3-regular Ising models with local field and interactions sampled from a standard normal. Each method in the graph has 10 points, each representing an Ising model of different size (*10 *to* 100 *variables).*

Our results suggest that by taking both long and short loops into account we can significantly improve the accuracy of inference at the cost of more computation time. In fact both plots in figure 2.5 show a log-log trend in time versus accuracy which suggests that taking short and long loops into account has almost independently improved the quality of inference.

2.5 Survey Propagation: semirings on semirings

Survey propagation (SP) was first introduced as a message passing solution to satisfiability [48] and was later generalized to general CSP [47] and arbitrary inference problems over factor-graphs [208]. Several works offer different interpretations and generalizations of survey propagation [46, 177, 199]. Here, we propose a generalization based the same notions that extends the application of BP to arbitrary commutative semirings. Our derivation closely follows and generalizes the variational approach of Mézard and Montanari [208], in the same way that the algebraic approach to BP (using commutative semirings) generalizes the variational derivation of sum-product and min-sum BP.

As a fixed point iteration procedure, if BP has more than one fixed points, it may not converge at all. Alternatively, if the messages are initialized properly BP may converge to one of its fixed points. SP equations, take "all" BP fixed points into account. In our algebraic perspective, this accounting of all fixed points is using a third operation $\widehat{\oplus}$. In particular, we require that $\otimes$ also distribute over $\widehat{\oplus}$, forming a second commutative semiring. We refer to the this new semiring as **SP semiring**.

Table 2.1: *The correspondence between BP and SP*

Belief Propagation	**Survey Propagation**
domain:	
$\underline{x}$	$\widehat{\underline{p}}_{\cdot\to\cdot}$
$\forall i \quad x_i$	$\widehat{p}_{i\to \mathrm{I}}$, $\widehat{p}_{\mathrm{I}\to i} \quad \forall i, \mathrm{I} \in \partial i$
$\mathcal{X}$	$\mathcal{P}$
expanded form:	
$q(\underline{x})$	$Q(\widehat{\underline{p}}_{\cdot\to\cdot})(\emptyset)$
integration:	
$q(\emptyset) = \bigoplus_{\underline{x}} q(\underline{x})$	$Q(\emptyset)(\emptyset) = \widehat{\bigoplus}_{\widehat{\underline{p}}_{\cdot\to\cdot}} Q(\widehat{\underline{p}}_{\cdot\to\cdot})(\emptyset)$
marginalization:	
$p(x_i) \propto \bigoplus_{\underline{x}\setminus i} p(\underline{x})$	$S(\widehat{p}_{\mathrm{I}\to i}) \propto \widehat{\bigoplus}_{\setminus \widehat{\underline{p}}_{\mathrm{I}\to i}} P(\widehat{\underline{p}}_{\cdot\to\cdot})$
factors:	
$\forall \mathrm{I} \quad f_{\mathrm{I}}(\underline{x}_{\mathrm{I}})$	$\widetilde{P}_{\mathrm{I}}(\widehat{\underline{p}}_{\partial \mathrm{I}\to \mathrm{I}})(\emptyset)$, $\widetilde{P}_i(\widehat{\underline{p}}_{\partial i\setminus \mathrm{I}\to i})(\emptyset)$ and $\widetilde{P}_{i\leftrightarrow \mathrm{I}}(\widehat{p}_{i\to \mathrm{I}}, \widehat{p}_{\mathrm{I}\to i})(\emptyset)^{-1} \quad \forall i, \mathrm{I} \in \partial i$

Let $\widehat{\underline{p}}_{\cdot\to\cdot}$ be a BP fixed point – that is

$$\widehat{\underline{p}}_{\cdot\to\cdot} = \{\forall i, \mathrm{I} \in \partial i \quad \widehat{p}_{i\to \mathrm{I}} = P_{i\to \mathrm{I}}(\widehat{\underline{p}}_{\partial i\setminus \mathrm{I}\to i}), \widehat{p}_{\mathrm{I}\to i} = P_{\mathrm{I}\to i}(\widehat{\underline{p}}_{\partial \mathrm{I}\setminus i\to \mathrm{I}})\}$$

and denote the set of all such fixed points by $\mathcal{P}$. Each BP fixed point corresponds to an approximation to the $q(\emptyset)$, which we denote by $Q(\widehat{\underline{p}}_{\cdot\to\cdot})(\emptyset)$ – using this functional form is to emphasize the dependence of this approximation on BP messages. Recall that in the original problem, $\mathcal{X}$ is the domain of assignments, $q(\underline{x})$ is the expanded form and $\oplus$-marginalization is (approximately) performed by BP. In the case of survey propagation, $\mathcal{P}$ is domain of assignments and the integral $Q(\widehat{\underline{p}}_{\cdot\to\cdot})(\emptyset)$ evaluates a particular assignment $\widehat{\underline{p}}_{\cdot\to\cdot}$ to all the messages – *i.e.*, $Q(\widehat{\underline{p}}_{\cdot\to\cdot})(\emptyset)$ is the new expanded form.

In this algebraic perspective, SP efficiently performs a second integral using $\widehat{\oplus}$ over all fixed points:

$$Q(\emptyset)(\emptyset) = \widehat{\bigoplus}_{\widehat{\underline{p}}_{\cdot\to\cdot} \in \mathcal{P}} Q(\underline{q}_{\cdot\to\cdot})(\emptyset) \tag{2.48}$$

Table 2.1 summarizes this correspondence.

Our derivation requires $(\mathcal{Y}^*, \otimes)$ to be an Abelian group (*i.e.*, every element of $\mathcal{Y}^*$ has an inverse w.r.t. $\otimes$). The requirement for invertablity of $\otimes$ is because we need to work with normalized BP and SP messages. In section 2.5.4 we introduce another variation of SP that simply counts the BP fixed points and relaxes this requirement.

2.5.1 Decomposition of the integral

In writing the normalized BP equations in section 2.1, we hid the normalization constant using $\propto$ sign. Here we explicitly define the normalization constants or **local integrals** by defining unnormalized messages, based on their normalized version

$$\widetilde{\mathrm{p}}_{\mathrm{I}\to i}(x_i) \stackrel{\text{def}}{=} \bigoplus_{\underline{x}_{\setminus i}} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \bigotimes_{j\in\partial \mathrm{I}\setminus i} \widehat{\mathrm{p}}_{j\to\mathrm{I}}(x_j) \stackrel{\text{def}}{=} \widetilde{\mathrm{P}}_{\mathrm{I}\to i}(\widehat{\underline{\mathrm{p}}}_{\partial \mathrm{I}\setminus i\to\mathrm{I}})(x_i) \tag{2.49}$$

$$\widetilde{\mathrm{p}}_{i\to\mathrm{I}}(x_i) \stackrel{\text{def}}{=} \bigotimes_{\mathrm{J}\in\partial i\setminus\mathrm{I}} \widehat{\mathrm{p}}_{\mathrm{J}\to i}(x_i) \stackrel{\text{def}}{=} \widetilde{\mathrm{P}}_{i\to\mathrm{I}}(\widehat{\underline{\mathrm{p}}}_{\partial i\setminus\mathrm{I}\to i})(x_i) \tag{2.50}$$

$$\widetilde{\mathrm{p}}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \stackrel{\text{def}}{=} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \bigotimes_{i\in\partial \mathrm{I}} \widehat{\underline{\mathrm{p}}}_{i\to\mathrm{I}}(x_i) \stackrel{\text{def}}{=} \widetilde{\mathrm{P}}_{\mathrm{I}}(\widehat{\underline{\mathrm{p}}}_{\partial \mathrm{I}\to\mathrm{I}})(\underline{x}_{\mathrm{I}}) \tag{2.51}$$

$$\widetilde{\mathrm{p}}_{i}(x_i) \stackrel{\text{def}}{=} \bigotimes_{\mathrm{I}\in\partial i} \widehat{\mathrm{p}}_{\mathrm{I}\to i}(x_i) \stackrel{\text{def}}{=} \widetilde{\mathrm{P}}_{i}(\widehat{\underline{\mathrm{p}}}_{\partial i\to i})(x_i) \tag{2.52}$$

where each update also has a functional form on the r.h.s. In each case, the local integrals are simply the integral of unnormalized messages or marginals – *e.g.*, $\widetilde{\mathrm{p}}_{\mathrm{I}\to i}(\emptyset) = \bigoplus_{x_i} \widetilde{\mathrm{p}}_{\mathrm{I}\to i}(x_i)$.

Define the functional $\widetilde{\mathrm{P}}_{i\leftrightarrow\mathrm{I}}(\widehat{\mathrm{p}}_{i\to\mathrm{I}}, \widehat{\mathrm{p}}_{\mathrm{I}\to i})$ as the product of messages from i to I and vice versa

$$\widetilde{\underline{\mathrm{p}}}_{i\leftrightarrow\mathrm{I}}(x_i) \stackrel{\text{def}}{=} \widehat{\mathrm{p}}_{i\to\mathrm{I}}(x_i) \otimes \widehat{\mathrm{p}}_{\mathrm{I}\to i}(x_i) \stackrel{\text{def}}{=} \widetilde{\mathrm{P}}_{i\leftrightarrow\mathrm{I}}(\widehat{\mathrm{p}}_{i\to\mathrm{I}}, \widehat{\mathrm{p}}_{\mathrm{I}\to i})(x_i) \tag{2.53}$$

Theorem 2.5.1. *If the factor-graph has no loops and $(\mathcal{Y}^*, \otimes)$ is an Abelian group, the global integral decomposes to local BP integrals as*

$$\mathrm{q}(\emptyset) = \bigotimes_{\mathrm{I}} \widetilde{\mathrm{p}}_{\mathrm{I}}(\emptyset) \bigotimes_{i} \widetilde{\mathrm{p}}_{i}(\emptyset) \left(\bigotimes_{i,\mathrm{I}\in\partial i} \widetilde{\underline{\mathrm{p}}}_{i\leftrightarrow\mathrm{I}}(\emptyset) \right)^{-1} \tag{2.54}$$

or in other words $\mathrm{q}(\emptyset) = \mathrm{Q}(\widehat{\underline{\mathrm{p}}}_{\cdot\to\cdot})(\emptyset)$ *where*

$$\mathrm{Q}(\widehat{\underline{\mathrm{p}}}_{\cdot\to\cdot})(\emptyset) = \bigotimes_{\mathrm{I}} \widetilde{\mathrm{P}}_{\mathrm{I}}(\widehat{\underline{\mathrm{p}}}_{\partial \mathrm{I}\to\mathrm{I}})(\emptyset) \bigotimes_{i} \widetilde{\mathrm{P}}_{i}(\widehat{\underline{\mathrm{p}}}_{\partial i\to i})(\emptyset) \left(\bigotimes_{i,\mathrm{I}\in\partial i} \widetilde{\mathrm{P}}_{i\leftrightarrow\mathrm{I}}(\widehat{\mathrm{p}}_{i\to\mathrm{I}}, \widehat{\mathrm{p}}_{\mathrm{I}\to i})(\emptyset) \right)^{-1} \tag{2.55}$$

Proof. For this proof we build a tree around an root node r that is connected to one factor. (Since the factor-graph is a tree such a node always exists.) Send BP messages from the leaves, up towards the root r and back to the leaves. Here, any message $\mathrm{q}_{i\to\mathrm{I}}(x_i)$, can give us the integral for the subtree that contains all the nodes and factors up to node i using $\mathrm{q}_{i\to\mathrm{I}}(\emptyset) = \bigoplus_{x_i} \mathrm{q}_{i\to\mathrm{I}}(x_i)$. Noting that the root is connected to exactly one factor, the global integral is

$$\bigoplus_{x_r} \mathrm{q}(x_r) = \bigoplus_{x_r} \bigotimes_{\mathrm{I}\in\partial r} \mathrm{q}_{\mathrm{I}\to r}(x_r) = \mathrm{q}_{\mathrm{I}\to r}(\emptyset) \tag{2.56}$$

On the other hand, We have the following relation between $q_{i\to I}$ and $\widehat{p}_{i\to I}$ (also corresponding factor-to-variable message)

$$q_{i\to I}(x_i) = \widehat{p}_{i\to I}(x_i) \otimes q_{i\to I}(\emptyset) \quad \forall i, I \in \partial i \tag{2.57}$$

$$q_{I\to i}(x_i) = \widehat{p}_{I\to i}(x_i) \otimes q_{I\to i}(\emptyset) \quad \forall i, I \in \partial i \tag{2.58}$$

Substituting this into BP equation (2.2) we get

$$q_{i\to I}(x_i) = \bigotimes_{J\in\partial i\setminus I} q_{J\to i}(\emptyset)\widehat{p}_{J\to i}(x_i) \tag{2.59}$$

$$q_{I\to i}(x_i) = \bigoplus_{\underline{x}_{\setminus i}} f_I(\underline{x}_I) \bigotimes_{j\in\partial I\setminus i} q_{j\to I}(\emptyset)\widehat{p}_{j\to I}(x_j) \tag{2.60}$$

By summing over both l.h.s and r.h.s in equations above and substituting from equation (2.50) we get

$$\begin{aligned} \bigoplus_{x_i} q_{i\to I}(x_i) &= \left(\bigotimes_{J\in\partial i\setminus I} q_{J\to i}(\emptyset)\right) \otimes \left(\bigoplus_{x_i} \bigotimes_{J\in\partial i\setminus I} \widehat{p}_{J\to i}(x_i)\right) \to \\ q_{i\to I}(\emptyset) &= \widetilde{p}_{i\to I}(\emptyset) \bigotimes_{J\in\partial i\setminus I} q_{J\to i}(\emptyset) \end{aligned} \tag{2.61}$$

and similarly for equation (2.60) using integration and substitution from equation (2.49) we have

$$\begin{aligned} \bigoplus_{x_i} q_{I\to i}(x_i) &= \left(\bigotimes_{j\in\partial I\setminus i} q_{j\to I}(\emptyset)\right) \otimes \left(\bigoplus_{\underline{x}_I} f_I(\underline{x}_I) \bigotimes_{j\in\partial I\setminus i} \widehat{p}_{j\to I}(x_j)\right) \to \\ q_{I\to i}(\emptyset) &= \widetilde{p}_{I\to i}(\emptyset) \bigotimes_{j\in\partial I\setminus i} q_{j\to I}(\emptyset) \end{aligned} \tag{2.62}$$

Equation (2.61) and 2.61 are simply recursive integration on a tree, where the integral up to node i (*i.e.*, $q_{i\to I}(\emptyset)$ in equation (2.61)) is reduced to integral in its sub-trees. By unrolling this recursion we see that $q_{i\to I}(\emptyset)$ is simply the product of all $\widetilde{p}_{I\to i}(\emptyset)$ and $\widetilde{p}_{I\to i}(\emptyset)$ in its sub-tree, where the messages are towards the root. Equation (2.56) tells us that the global integral is not different. Therefore, equation (2.61) we can completely expand the recursion for the global integral. For this, let $\uparrow i$ restrict the ∂i to the factor that is higher than variable i in the tree (*i.e.*, closer to the root r). Similarly let $\uparrow I$ be the variable that is closer to the root than I. We can write the global integral as

$$q(\emptyset) = \bigotimes_{i, I=\uparrow i} \widetilde{p}_{i\to I}(\emptyset) \bigotimes_{I, i=\uparrow I} \widetilde{p}_{I\to i}(\emptyset) \tag{2.63}$$

Proposition 2.5.2 shows that these local integrals can be written in terms of local integrals of

interest – *i.e.*,

$$\widetilde{\mathrm{p}}_{\mathrm{I}\to i}(\emptyset) \quad = \quad \frac{\widetilde{\mathrm{p}}_{\mathrm{I}}(\emptyset)}{\widetilde{\underline{\mathrm{p}}}_{i\leftrightarrow \mathrm{I}}(\emptyset)} \quad \text{and} \quad \widetilde{\mathrm{p}}_{i\to \mathrm{I}}(\emptyset) \quad = \quad \frac{\widetilde{\mathrm{p}}_{i}(\emptyset)}{\widetilde{\underline{\mathrm{p}}}_{i\leftrightarrow \mathrm{I}}(\emptyset)}$$

Substituting from the equations above into equation (2.63) we get the equations of theorem 2.5.1.

□

Proof. (proposition 2.5.2 on page 63) By definition of $\widetilde{\mathrm{p}}_{\mathrm{I}}(\underline{x}_{\mathrm{I}})$ and $\widehat{\mathrm{p}}_{i\to \mathrm{I}}(x_i)$ in equation (2.49)

$$\widetilde{\mathrm{p}}_{\mathrm{I}}(x_i) = \widetilde{\mathrm{p}}_{\mathrm{I}\to i}(x_i) \otimes \widehat{\mathrm{p}}_{i\to \mathrm{I}}(x_i) \quad \to \quad \bigoplus\nolimits_{x_i} \widetilde{\mathrm{p}}_{\mathrm{I}}(x_i) = \bigoplus\nolimits_{x_i} \widetilde{\mathrm{p}}_{\mathrm{I}\to i}(x_i) \otimes \widehat{\mathrm{p}}_{i\to \mathrm{I}}(x_i) \quad \to$$
$$\widetilde{\mathrm{p}}_{\mathrm{I}}(\emptyset) = \widetilde{\mathrm{p}}_{\mathrm{I}\to i}(\emptyset) \otimes \left(\bigoplus\nolimits_{x_i} \widehat{\mathrm{p}}_{\mathrm{I}\to i}(x_i) \otimes \widehat{\mathrm{p}}_{i\to \mathrm{I}}(x_i)\right) \quad \to \quad \widetilde{\mathrm{p}}_{\mathrm{I}}(\emptyset) = \widetilde{\mathrm{p}}_{\mathrm{I}\to i}(\emptyset) \otimes \widetilde{\underline{\mathrm{p}}}_{i\leftrightarrow \mathrm{I}}(\emptyset)$$

where in the last step we used equation (2.53).

Similarly for the second statement of the proposition we have

$$\widetilde{\mathrm{p}}_{i}(x_i) = \widetilde{\mathrm{p}}_{i\to \mathrm{I}}(x_i) \otimes \widehat{\mathrm{p}}_{\mathrm{I}\to i}(x_i) \quad \to \quad \bigoplus\nolimits_{x_i} \widetilde{\mathrm{p}}_{i}(x_i) = \bigoplus\nolimits_{x_i} \widetilde{\mathrm{p}}_{i\to \mathrm{I}}(x_i) \otimes \widehat{\mathrm{p}}_{\mathrm{I}\to i}(x_i) \quad \to$$
$$\widetilde{\mathrm{p}}_{i}(\emptyset) = \widetilde{\mathrm{p}}_{i\to \mathrm{I}}(\emptyset) \otimes \left(\bigoplus\nolimits_{x_i} \widehat{\mathrm{p}}_{\mathrm{I}\to i}(x_i) \otimes \widehat{\mathrm{p}}_{i\to \mathrm{I}}(x_i)\right) \quad \to \quad \widetilde{\mathrm{p}}_{i}(\emptyset) = \widetilde{\mathrm{p}}_{i\to \mathrm{I}}(\emptyset) \otimes \widetilde{\underline{\mathrm{p}}}_{i\leftrightarrow \mathrm{I}}(\emptyset)$$

□

2.5.2 The new factor-graph and semiring

The decomposition of integral in theorem 2.5.1 means $\mathrm{Q}(\widehat{\underline{\mathrm{p}}}_{\cdot\to\cdot})(\emptyset)$ has a factored form. Therefore, a factor-graph with $\widehat{\underline{\mathrm{p}}}_{\cdot\to\cdot}$ as the set of variables and three different types of factors corresponding to different terms in the decomposition – *i.e.*, $\widetilde{\mathrm{P}}_{\mathrm{I}}(\widehat{\underline{\mathrm{p}}}_{\partial \mathrm{I}\to \mathrm{I}})(\emptyset)$, $\widetilde{\mathrm{P}}_{i}(\widehat{\underline{\mathrm{p}}}_{\partial i\setminus \mathrm{I}\to i})(\emptyset)$ and $\widetilde{\mathrm{P}}_{i\leftrightarrow \mathrm{I}}(\widehat{\mathrm{p}}_{i\to \mathrm{I}}, \widehat{\mathrm{p}}_{\mathrm{I}\to i})(\emptyset)^{-1}$ can represent $\mathrm{Q}(\widehat{\underline{\mathrm{p}}}_{\cdot\to\cdot})(\emptyset)$.

Figure 2.6 shows a simple factor-graph and the corresponding SP factor-graph. The new factor-graph has one variable per each message in the original factor-graph and three types of factors as discussed above. Survey propagation is simply belief propagation applied to the this new factor-graph using the new semiring. As before BP messages are exchanged between variables and factors. But here, we can simplify BP messages by substitution and only keep two types of factor-to-factor messages. We use $\mathrm{S}_{i\to \mathrm{I}}$ and $\mathrm{S}_{\mathrm{I}\to i}$ to denote these two types of SP messages. These messages are exchanged between two types of factors, namely $\widetilde{\mathrm{P}}_{\mathrm{I}}(\widehat{\underline{\mathrm{p}}}_{\partial \mathrm{I}\to \mathrm{I}})(\emptyset)$ and $\widetilde{\mathrm{P}}_{i}(\widehat{\underline{\mathrm{p}}}_{\partial i\setminus \mathrm{I}\to i})(\emptyset)$. Since the third type of factors $\widetilde{\mathrm{P}}_{i\leftrightarrow \mathrm{I}}(\widehat{\mathrm{p}}_{i\to \mathrm{I}}, \widehat{\mathrm{p}}_{\mathrm{I}\to i})(\emptyset)^{-1}$ are always connected to only two variables, $\widehat{\mathrm{p}}_{i\to \mathrm{I}}$ and

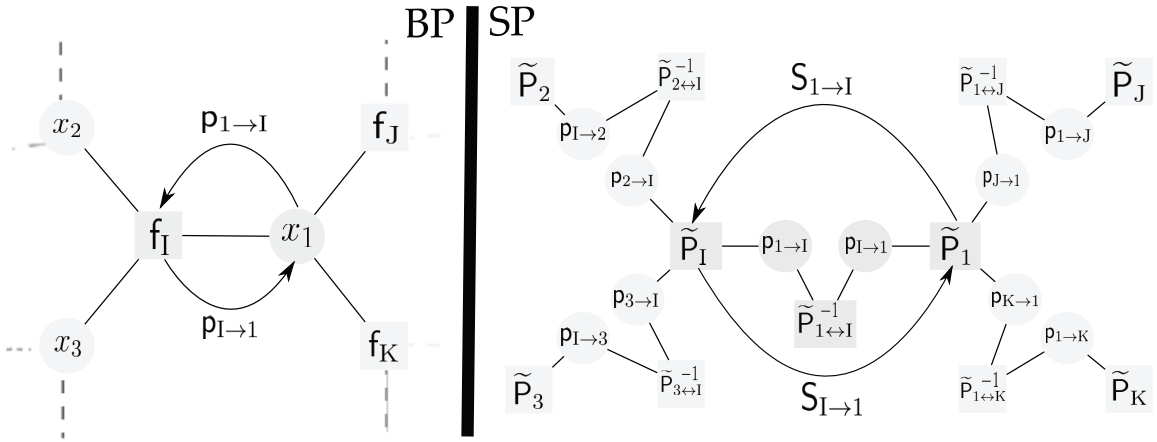


Figure 2.6: *Part of a factor-graph (left) and the corresponding SP factor-graph on the right. The variables in SP factor-graph are the messages in the original graph. The SP factor-graph has three type of factors: (I)* $\widetilde{\mathsf{P}}_\mathrm{I}(.)(\emptyset)$*, (II)* $\widetilde{\mathsf{P}}_i(.)(\emptyset)$ *and (III)* $\widetilde{\mathsf{P}}_{i\leftrightarrow\mathrm{I}}(.)(\emptyset)^{-1}$*. As the arrows suggest, SP message updates are simplified so that only two type of messages are exchanged:* $\mathsf{S}_{i\to\mathrm{I}}$ *and* $\mathsf{S}_{\mathrm{I}\to i}$ *between factors of type (I) and (II).*

$\widehat{\mathsf{p}}_{\mathrm{I}\to i}$, we can simplify their role in the SP message update to get

$$\mathsf{S}_{i\to\mathrm{I}}(\widehat{\mathsf{p}}_{i\to\mathrm{I}},\widehat{\mathsf{p}}_{\mathrm{I}\to i}) \;\propto\; \widehat{\bigoplus}_{\setminus\widehat{\mathsf{p}}_{i\to\mathrm{I}},\widehat{\mathsf{p}}_{\mathrm{I}\to i}} \left(\frac{\widetilde{\mathsf{P}}_i(\underline{\widehat{\mathsf{p}}}_{\partial i\to i})(\emptyset)}{\widetilde{\mathsf{P}}_{i\leftrightarrow\mathrm{I}}(\widehat{\mathsf{p}}_{i\to\mathrm{I}},\widehat{\mathsf{p}}_{\mathrm{I}\to i})(\emptyset)} \bigotimes_{\mathrm{J}\in\partial i\setminus\mathrm{I}} \mathsf{S}_{\mathrm{J}\to i}(\widehat{\mathsf{p}}_{i\to\mathrm{J}},\widehat{\mathsf{p}}_{\mathrm{J}\to i}) \right) \tag{2.64}$$

$$\mathsf{S}_{\mathrm{I}\to i}(\widehat{\mathsf{p}}_{i\to\mathrm{I}},\widehat{\mathsf{p}}_{\mathrm{I}\to i}) \;\propto\; \widehat{\bigoplus}_{\setminus\widehat{\mathsf{p}}_{i\to\mathrm{I}},\widehat{\mathsf{p}}_{\mathrm{I}\to i}} \left(\frac{\widetilde{\mathsf{P}}_\mathrm{I}(\underline{\widehat{\mathsf{p}}}_{\partial\mathrm{I}\to\mathrm{I}})(\emptyset)}{\widetilde{\mathsf{P}}_{i\leftrightarrow\mathrm{I}}(\widehat{\mathsf{p}}_{i\to\mathrm{I}},\widehat{\mathsf{p}}_{\mathrm{I}\to i})(\emptyset)} \bigotimes_{j\in\partial\mathrm{I}\setminus i} \mathsf{S}_{j\to\mathrm{I}}(\widehat{\mathsf{p}}_{j\to\mathrm{I}},\widehat{\mathsf{p}}_{\mathrm{I}\to j}) \right) \tag{2.65}$$

where in all cases we are assuming the messages $\underline{\widehat{\mathsf{p}}}_{\cdot\to\cdot} \in \mathcal{P}$ are consistent with each other – *i.e.*, satisfy BP equations on the original factor-graph. Note that, here again we are using the normalized BP message update and the normalization factor is hidden using $\propto$ sign. This is possible because we assumed $\otimes$ has an inverse. We can further simplify this update using the following proposition.

Proposition 2.5.2. *for* $\underline{\widehat{\mathsf{p}}}_{\cdot\to\cdot} \in \mathcal{P}$

$$\frac{\widetilde{\mathsf{P}}_i(\underline{\widehat{\mathsf{p}}}_{\partial i\to i})(\emptyset)}{\widetilde{\mathsf{P}}_{i\leftrightarrow\mathrm{I}}(\widehat{\mathsf{p}}_{\mathrm{I}\to i},\widehat{\mathsf{p}}_{i\to\mathrm{I}})(\emptyset)} \;=\; \widetilde{\mathsf{P}}_{i\to\mathrm{I}}(\underline{\widehat{\mathsf{p}}}_{\partial i\setminus\mathrm{I}\to i})(\emptyset) \tag{2.66}$$

and

$$\frac{\widetilde{\mathsf{P}}_\mathrm{I}(\underline{\widehat{\mathsf{p}}}_{\partial\mathrm{I}\to\mathrm{I}})(\emptyset)}{\widetilde{\mathsf{P}}_{i\leftrightarrow\mathrm{I}}(\widehat{\mathsf{p}}_{\mathrm{I}\to i},\widehat{\mathsf{p}}_{i\to\mathrm{I}})(\emptyset)} \;=\; \widetilde{\mathsf{P}}_{\mathrm{I}\to i}(\underline{\widehat{\mathsf{p}}}_{\partial\mathrm{I}\setminus i\to\mathrm{I}})(\emptyset) \tag{2.67}$$

Proof. By definition of $\widetilde{\mathrm{p}}_{\mathrm{I}}(\underline{x}_{\mathrm{I}})$ and $\widehat{\mathrm{p}}_{i\to\mathrm{I}}(x_i)$ in equation (2.49)

$$\widetilde{\mathrm{p}}_{\mathrm{I}}(x_i) = \widetilde{\mathrm{p}}_{\mathrm{I}\to i}(x_i)\otimes\widehat{\mathrm{p}}_{i\to\mathrm{I}}(x_i) \quad\to\quad \bigoplus_{x_i}\widetilde{\mathrm{p}}_{\mathrm{I}}(x_i) = \bigoplus_{x_i}\widetilde{\mathrm{p}}_{\mathrm{I}\to i}(x_i)\otimes\widehat{\mathrm{p}}_{i\to\mathrm{I}}(x_i) \quad\to$$
$$\widetilde{\mathrm{p}}_{\mathrm{I}}(\emptyset) = \widetilde{\mathrm{p}}_{\mathrm{I}\to i}(\emptyset)\otimes\Big(\bigoplus_{x_i}\widehat{\mathrm{p}}_{\mathrm{I}\to i}(x_i)\otimes\widehat{\mathrm{p}}_{i\to\mathrm{I}}(x_i)\Big) \quad\to\quad \widetilde{\mathrm{p}}_{\mathrm{I}}(\emptyset) = \widetilde{\mathrm{p}}_{\mathrm{I}\to i}(\emptyset)\otimes\widetilde{\underline{\mathrm{p}}}_{i\leftrightarrow\mathrm{I}}(\emptyset)$$

where in the last step we used equation (2.53).

Similarly for the second statement of the proposition we have

$$\widetilde{\mathrm{p}}_{i}(x_i) = \widetilde{\mathrm{p}}_{i\to\mathrm{I}}(x_i)\otimes\widehat{\mathrm{p}}_{\mathrm{I}\to i}(x_i) \quad\to\quad \bigoplus_{x_i}\widetilde{\mathrm{p}}_{i}(x_i) = \bigoplus_{x_i}\widetilde{\mathrm{p}}_{i\to\mathrm{I}}(x_i)\otimes\widehat{\mathrm{p}}_{\mathrm{I}\to i}(x_i) \quad\to$$
$$\widetilde{\mathrm{p}}_{i}(\emptyset) = \widetilde{\mathrm{p}}_{i\to\mathrm{I}}(\emptyset)\otimes\Big(\bigoplus_{x_i}\widehat{\mathrm{p}}_{\mathrm{I}\to i}(x_i)\otimes\widehat{\mathrm{p}}_{i\to\mathrm{I}}(x_i)\Big) \quad\to\quad \widetilde{\mathrm{p}}_{i}(\emptyset) = \widetilde{\mathrm{p}}_{i\to\mathrm{I}}(\emptyset)\otimes\widetilde{\underline{\mathrm{p}}}_{i\leftrightarrow\mathrm{I}}(\emptyset)$$

□

The term on the l.h.s. in the proposition above appear in equation (2.64) and the terms on the r.h.s are local message integrals given by equation (2.49). We can enforce $\widehat{\underline{\mathrm{p}}}_{\cdot\to\cdot}\in\mathcal{P}$, by enforcing BP updates $\widehat{\mathrm{p}}_{i\to\mathrm{I}} = \mathrm{P}_{i\to\mathrm{I}}(\widehat{\underline{\mathrm{p}}}_{\partial i\setminus\mathrm{I}\to i})$ and $\widehat{\mathrm{p}}_{\mathrm{I}\to i} = \mathrm{P}_{\mathrm{I}\to i}(\widehat{\underline{\mathrm{p}}}_{\partial\mathrm{I}\setminus\to\mathrm{I}})$ "locally", during the message updates in the new factor-graph. Combining this constraint with the simplification offered by proposition 2.5.2 gives us the SP message updates

$$\mathrm{S}_{i\to\mathrm{I}}(\widehat{\mathrm{p}}_{i\to\mathrm{I}}) \propto \widehat{\bigoplus_{\widehat{\underline{\mathrm{p}}}_{\partial i\setminus\mathrm{I}\to i}}}\Big(1\big(\widehat{\mathrm{p}}_{i\to\mathrm{I}} = \mathrm{P}_{i\to\mathrm{I}}(\widehat{\underline{\mathrm{p}}}_{\partial i\setminus\mathrm{I}\to i})\big)\otimes\mathrm{P}_{i\to\mathrm{I}}(\widehat{\underline{\mathrm{p}}}_{\partial i\setminus\mathrm{I}\to i})(\emptyset)\bigotimes_{\mathrm{J}\in\partial i\setminus\mathrm{I}}\mathrm{S}_{\mathrm{J}\to i}(\widehat{\mathrm{p}}_{\mathrm{J}\to i})\Big) \tag{2.68}$$

$$\mathrm{S}_{\mathrm{I}\to i}(\widehat{\mathrm{p}}_{\mathrm{I}\to i}) \propto \widehat{\bigoplus_{\widehat{\underline{\mathrm{p}}}_{\partial\mathrm{I}\setminus i\to\mathrm{I}}}}\Big(1\big(\widehat{\mathrm{p}}_{\mathrm{I}\to i} = \mathrm{P}_{\mathrm{I}\to i}(\widehat{\underline{\mathrm{p}}}_{\partial\mathrm{I}\setminus i\to\mathrm{I}})\big)\otimes\mathrm{P}_{\mathrm{I}\to i}(\widehat{\underline{\mathrm{p}}}_{\partial\mathrm{I}\setminus i\to\mathrm{I}})(\emptyset)\bigotimes_{j\in\partial\mathrm{I}\setminus i}\mathrm{S}_{j\to\mathrm{I}}(\widehat{\mathrm{p}}_{j\to\mathrm{I}})\Big) \tag{2.69}$$

where $1(.)$ is the identity function on the SP semiring, where $1(\textsc{true}) = \widehat{1}$ and $1(\textsc{false}) = \widehat{0}$.

Here each SP message is a functional over all possible BP messages between the same variable and factor. However, in updating the SP messages, the identity functions ensure that only the messages that locally satisfy BP equations are taken into account. Another difference from the updates of equation (2.64) is that SP messages have a single argument. This is because the new local integrals either depend on $\widehat{\mathrm{p}}_{i\to\mathrm{I}}$ or $\widehat{\mathrm{p}}_{\mathrm{I}\to i}$, and not both.

Example 2.5.1. In variational approach, survey propagation comes in two variations: entropic SP(ξ) and energetic SP(y) [208]. For the readers familiar with variational derivation of SP, here we express the relation to the algebraic approach. According to the variational view, the partition function of the *entropic SP* is $\sum_{\widehat{\underline{\mathrm{p}}}_{\cdot\to\cdot}} e^{\xi\log(\mathrm{Q}(\widehat{\underline{\mathrm{p}}}_{\cdot\to\cdot})(\emptyset))}$, where $\mathrm{Q}(\widehat{\underline{\mathrm{p}}}_{\cdot\to\cdot})(\emptyset)$ is the partition function for the sum-product semiring. The entropic SP has an inverse temperature parameter, a.k.a. *Parisi parameter*, $\xi\in\mathbb{R}$. It is easy to see that $\xi = 1$ corresponds to $\widehat{\oplus} = +, \oplus = +$ and $\otimes = \times$ in our algebraic approach. The limits of $\xi\to\infty$ corresponds to $\widehat{\oplus} = \max$. On the other hand, the limit

of $\xi \to 0$ amounts to ignoring $\mathrm{Q}(\widehat{\underline{\mathrm{p}}}_{\cdot\to\cdot})(\emptyset)$ and corresponds to the counting SP; see section 2.5.4.

The *energetic* $\mathrm{SP}(\mathbf{y})$ is different only in the sense that $\mathrm{Q}(\widehat{\underline{\mathrm{p}}}_{\cdot\to\cdot})(\emptyset)$ in $\sum_{\widehat{\underline{\mathrm{p}}}_{\cdot\to\cdot}} e^{-\mathbf{y}\log(\mathrm{Q}(\widehat{\underline{\mathrm{p}}}_{\cdot\to\cdot})(\emptyset))}$ is the ground state energy. This corresponds to $\widehat{\oplus} = +, \oplus = \max$ and $\otimes = \sum$, and the limits of the inverse temperature parameter $\mathbf{y} \to \infty$ is equivalent to $\widehat{\oplus} = \min, \oplus = \min$ and $\otimes = \sum$. By taking an algebraic view we can choose between both operations and domains. For instance, an implication of algebraic view is that all the variations of SP can be applied to the domain of complex numbers $\mathcal{Y}^* = \mathbb{C}$.

2.5.3 The new integral and marginals

Once again we can use theorem 2.5.1, this time to approximate the *SP integral* $\mathrm{Q}(\emptyset)(\emptyset) = \widehat{\bigoplus}_{\widehat{\underline{\mathrm{p}}}_{\cdot\to\cdot}} \mathrm{Q}(\widehat{\underline{\mathrm{p}}}_{\cdot\to\cdot})(\emptyset)$ using local integral of SP messages.

The *SP marginal* over each BP message $\widehat{\mathrm{p}}_{i\to\mathrm{I}}$ or $\widehat{\mathrm{p}}_{\mathrm{I}\to i}$ is the same as the corresponding SP message – *i.e.*, $\mathrm{S}(\widehat{\mathrm{p}}_{i\to\mathrm{I}}) = \mathrm{S}_{i\to\mathrm{I}}(\widehat{\mathrm{p}}_{i\to\mathrm{I}})$. To see this in the factor-graph of figure 2.6, note that each message variable is connected to two factors, and both of these factors are already contained in calculating one SP messages.

Moreover, from the SP marginals over messages we can recover the SP marginals over BP marginals which we denote by $\mathrm{S}(\widehat{\mathrm{p}})(x_i)$. For this, we simply need to enumerate all combinations of BP messages that produce a particular marginal

$$\mathrm{S}(\widehat{\mathrm{p}})(x_i) \propto \widehat{\bigoplus}_{\widehat{\underline{\mathrm{p}}}_{\partial i\to i}} 1(\widehat{\mathrm{p}}(x_i) = \mathrm{P}(\widehat{\underline{\mathrm{p}}}_{\partial i\to i})(x_i)) \bigotimes_{\mathrm{I}\in\partial i} \mathrm{S}_{\mathrm{I}\to i}(\widehat{\mathrm{p}}_{\mathrm{I}\to i}) \tag{2.70}$$

2.5.4 Counting survey propagation

Previously we required the $\otimes$ operator to have an inverse, so that we can decompose the BP integral $\mathrm{q}(\emptyset)$ into local integrals. Moreover, for a consistent decomposition of the BP integral, SP and BP semiring previously shared the $\otimes$ operation.[8]

Here, we lift these requirements by discarding the BP integrals altogether. This means SP semiring could be completely distinct from BP semiring and $(\mathcal{Y}^*, \otimes)$ does not have to be an Abelian group. This setting is particularly interesting when the SP semiring is sum-product over real

[8] This is because if the expansion operation $\widehat{\otimes}$ was different from the expansion operation of BP, $\otimes$, the expanded form $\mathrm{Q}(\widehat{\underline{\mathrm{p}}}_{\cdot\to\cdot})$ in the SP factor-graph would not evaluate the integral $\mathrm{q}(\emptyset)$ in the BP factor-graph, even in factor-graphs without any loops.

domain

$$\mathsf{S}_{i\to\mathrm{I}}(\widehat{\mathsf{p}}_{i\to\mathrm{I}}) \propto \sum_{\widehat{\underline{\mathsf{p}}}_{\partial i\setminus\mathrm{I}\to i}} 1\Big(\widehat{\mathsf{p}}_{i\to\mathrm{I}} = \mathsf{P}_{i\to\mathrm{I}}(\widehat{\underline{\mathsf{p}}}_{\partial i\setminus\mathrm{I}\to i})\Big) \prod_{\mathrm{J}\in\partial i\setminus\mathrm{I}} \mathsf{S}_{\mathrm{J}\to i}(\widehat{\mathsf{p}}_{\mathrm{J}\to i}) \tag{2.71}$$

$$\mathsf{S}_{\mathrm{I}\to i}(\widehat{\mathsf{p}}_{\mathrm{I}\to i}) \propto \sum_{\widehat{\underline{\mathsf{p}}}_{\partial\mathrm{I}\setminus i\to\mathrm{I}}} 1\Big(\widehat{\mathsf{p}}_{\mathrm{I}\to i} = \mathsf{P}_{\mathrm{I}\to i}(\widehat{\underline{\mathsf{p}}}_{\partial\mathrm{I}\setminus i\to\mathrm{I}})\Big) \prod_{j\in\partial\mathrm{I}\setminus i} \mathsf{S}_{j\to\mathrm{I}}(\widehat{\mathsf{p}}_{j\to\mathrm{I}}) \tag{2.72}$$

Here, the resulting SP integral $\mathsf{Q}(\widehat{\underline{\mathsf{p}}}_{\cdot\to\cdot}) = \sum_{\widehat{\underline{\mathsf{p}}}_{\cdot\to\cdot}} 1(\widehat{\underline{\mathsf{p}}}_{\cdot\to\cdot} \in \mathcal{P})$ simply "counts" the number of BP fixed points and SP marginals over BP marginals (given by equation (2.70)) approximates the frequency of a particular marginal. The original survey propagation equations in [48], that are very successful in solving satisfiability correspond to counting SP applied to the or-and semiring.

Example 2.5.2. Interestingly, in all min-max problems with discrete domains $\mathcal{X}$, min-max BP messages can only take the values that are in the range of factors – *i.e.*, $\mathcal{Y}^* = \mathcal{Y}$. This is because any ordered set is closed under min and max operations. Here, each counting SP message $\mathsf{S}_{i\to\mathrm{I}}(\widehat{\mathsf{p}}_{i\to\mathrm{I}}) : \mathcal{Y}^{|\mathcal{X}_i|} \to \mathbb{R}$ is a discrete distribution over all possible min-max BP messages. This means counting survey propagation where the BP semring is min-max is computationally "tractable". In contrast (counting) SP, when applied to sum-product BP over real domains is not tractable. This is because in this case each SP message is a distribution over a uncountable set: $\mathsf{S}_{i\to\mathrm{I}}(\widehat{\mathsf{p}}_{i\to\mathrm{I}}) : \mathbb{R}^{|\mathcal{X}_i|} \to \mathbb{R}$.

In practice, (counting) SP is only interesting if it remains tractable. The most well-known case corresponds to counting SP when applied to the or-and semiring. In this case the factors are constraints and the domain of SP messages is $\{\textsc{true}, \textsc{false}\}^{|\mathcal{X}_i|}$. Our algebraic perspective extends this set of tractable instances. For example, it show that counting SP can be used to count the number of fixed points of BP when applied to xor-and or min-max semiring.

2.6 Messages and particles

The contrasting properties of stochastic and deterministic approximations make a general hybrid method desirable. After reviewing the basics of MCMC in section 2.6.1, we discuss some particle-based approaches to message passing and introduce our hybrid inference method that combines message passing and Gibbs sampling in section 2.6.3. The discussions of this section are limited to sum-product inference.

2.6.1 Markov Chain Monte Carlo

Markov Chain Monte Carlo (MCMC) is a technique to produce samples from a target distribution p, by exploring a Markov Chain which is constructed such that more probable areas are visited more often [11, 227, 263].

A Markov Chain is a stochastic process $\underline{x}^{(0)}, \ldots, \underline{x}^{(t)}$ in which:

$$\mathrm{p}(\underline{x}^{(t)} \mid \underline{x}^{(t-1)}, \ldots, \underline{x}^{(1)}) \quad = \quad \mathrm{k}_t(\underline{x}^{(t)}, \underline{x}^{(t-1)}) \tag{2.73}$$

that is the current state $\underline{x}^{(t)}$ is independent of all the history, given only the previous state $\underline{x}^{(t-1)}$.

For a homogeneous Markov chain, the **transition kernel** $\mathrm{k}_t(\underline{x}^{(t)}, \underline{x}^{(t-1)})$ is the same for all t. In this case and under some assumptions[9], starting from any arbitrary distribution $\underline{x}^{(0)} \sim \mathrm{p}^{(0)}(\underline{x})$ after at least T_{mix} transitions by the chain, we have $\underline{x}^{(T_{\mathrm{mix}})} \sim \mathrm{p}(x)$. Given a set of particles $\underline{x}[1], \ldots, \underline{x}[L]$ sampled from $\mathrm{p}(\underline{x})$, we can estimate the marginal probabilities (or any other expectation) as

$$\widehat{\mathrm{p}}(x_i) \quad \propto \quad \frac{1}{L} \sum_{n=1}^{L} 1(x[n]_i = x_i) \tag{2.74}$$

For a given transition kernel, the following condition, known as detailed balance, identifies the stationary distribution p:

$$\begin{aligned} \mathrm{p}(\underline{x}^{(t)})\mathrm{k}(\underline{x}^{(t)}, \underline{x}^{(t-1)}) \quad &= \quad \mathrm{p}(\underline{x}^{(t-1)})\mathrm{k}(\underline{x}^{(t-1)}, \underline{x}^{(t)}) \quad \Rightarrow \\ \mathrm{p}(\underline{x}^{(t)}) \quad &= \quad \sum_{\underline{x}} \mathrm{p}(\underline{x})\ \mathrm{k}(\underline{x}^{(t)}; \underline{x}) \end{aligned} \tag{2.75}$$

which means that $\mathrm{p}(.)$ is the left eigenvector of $\mathrm{k}(.,.)$ with eigenvalue 1. All the other eigenvalues are less than one and the **mixing time**, T_{mix}, of the chain depends on the second largest eigenvalue; the smaller it is, the faster consecutive transition by $\mathrm{k}(.,.)$ shrinks the corresponding components, retaining only p.

Metropolis-Hasting Algorithm and Gibbs sampling

Many important MCMC algorithms can be interpreted as a special case of Metropolis-Hasting (MH) [124, 207]. Similar to importance sampling, MH uses proposal distribution $\mathrm{m}(\underline{x}^{(t)} \mid \underline{x}^{(n-1)})$, but in this case, the proposal distribution is to help with the design of transition kernel $\mathrm{k}(.,.)$. After sampling $\underline{x}^{(t)}$ from the proposal $\underline{x}^{(t)} \sim \mathrm{m}(\underline{x})$, it is accepted with probability

$$\min\left\{1\ ,\ \frac{\mathrm{p}(\underline{x}^{(t)})/\mathrm{p}(\underline{x}^{(t-1)})}{\mathrm{m}(\underline{x}^{(t)} \mid \underline{x}^{(t-1)})/\mathrm{m}(\underline{x}^{(t-1)} \mid \underline{x}^{(t)})}\right\} \tag{2.76}$$

where, if the proposed sample is not accepted, $\underline{x}^{(t)} = \underline{x}^{(t-1)}$.

The kernel resulting from this procedure admits the detailed balance condition w.r.t. the stationary distribution $\mathrm{p}(\underline{x})$. An important feature of MCMC, which allows for its application in

[9] The assumptions are: (I) Irreducibility: There is a non-zero probability of reaching all states starting with any arbitrary state and (II) Aperiodicity: The chain does not trap in cycles.

graphical models, is the possibility of building valid transition kernels as the **mixtures and cycles** of other transition kernels. If p, is the stationary distribution for k_1 and k_2, then it is also the stationary distribution for $k_1 k_2$ (cycle) and $\lambda k_1 + (1-\lambda)k_2, \quad 0 \leq \lambda \leq 1$ (mixture) [295].

Cycling of kernels gives us **Gibbs sampling** in graphical models [109], when kernels are

$$k_i(x_i^{(t)}, \underline{x}_{\Delta i}^{(t-1)}) \quad = \quad p(x_i^{(t)} \mid \underline{x}_{\Delta i}^{(t-1)}) \quad \forall i \tag{2.77}$$

where as before Δi is the Markov blanket of node i. It is also possible to use block MH-kernels with graphical models. In MH-samplers, when highly correlated variables are blocked together, mixing properties improve. In fact, the Gibbs sampler is such a method, with the proposal distribution that results in acceptance with probability 1.

Similar to the general MH-methods, Gibbs sampling can fail if the kernel does not mix properly. This could happen if variables are strongly correlated. In principle one can assemble neighbouring variables into blocks and update them as one [147]. However in difficult regimes the number of variables that should be flipped to move from one local optima to another, is in the order of total number of variables [208], which makes this approach intractable.

Mixture of kernels can be used to combine a global proposal with a local proposal (*e.g.*, [81, 88]). In fact if we could view a message passing operator as a transition kernel (at least when message passing is exact), then the mixture of kernels – *e.g.*, with Gibbs sampling – could produce interesting hybrid methods. In section 2.6.3, by combining BP and Gibbs sampling operator (when rephrased as message update) we introduce a new hybrid method.

2.6.2 Hybrid methods

Stochastic methods are slow in convergence but they are guaranteed to converge. Even if the kernel is reducible, samples will cover a subset of the true support – *i.e.*, MCMC still converges to a single sub-measure when the Gibbs measure is not unique.

On the other hand, deterministic approximations are fast but non-convergent in difficult regimes. Modifications that result in convergence are either generally intractable (*e.g.*, SP), slow (*e.g.*, loop corrections and the methods that tighten a bound over the free energy) and/or degrade the quality of solutions.

Moreover, sampling methods are flexible in representing distributions. This has motivated growing interest in nonparametric approach to variational inference [110] and in particular variations of Belief Propagation [141, 142, 144, 172, 231, 282, 283]. However, in the sense that these methods do not rely on a Markov chain for inference, they are closer to variational inference than MCMC methods.

To better appreciate this distinction, consider two closely related methods: Gibbs Sampling

and **hard-spin mean-field** [10], that uses the following update equation

$$\widehat{\mathrm{p}}(x_i) \quad \propto \quad \sum_{\underline{x}_{\setminus i}} \prod_{\mathrm{I}\in\partial i} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}\setminus i}, x_i) \prod_{j\in\mathrm{I}\setminus i} \widehat{\mathrm{p}}(x_j)$$

Interestingly, the **detailed balance** condition of equation (2.75) for Gibbs sampler gives us the same equation:

$$\widehat{\mathrm{p}}(x_i) \quad \propto \quad \sum_{\underline{x}_{\setminus i}} \prod_{\mathrm{I}\in\partial i} \widehat{\mathrm{p}}(x_i \mid \underline{x}_{\mathrm{I}\setminus i}) \prod_{j\in\mathrm{I}\setminus i} \widehat{\mathrm{p}}(x_j)$$

However, given enough iterations, Gibbs Sampling can be much more accurate than hard-spin mean field method. Here the difference is that, with Gibbs sampler, this equation is enforced by the chain rather than explicit averaging of distributions, which means the correlation information is better taken into account.

2.6.3 Perturbed Belief Propagation

Consider a single particle $\underline{\hat{x}} = \underline{x}[1]$ in Gibbs Sampling. At any time-step t, $\hat{x}_i$ is updated according to

$$\hat{x}_i^{(\mathrm{t})} \sim \widehat{\mathrm{p}}(x_i) \quad \propto \quad \prod_{\mathrm{I}\in\partial i} \mathrm{f}_{\mathrm{I}}(x_i, \underline{\hat{x}}_{\partial I\setminus i}^{(\mathrm{t}-1)}) \tag{2.78}$$

Here we establish a correspondence between a particle in Gibbs Sampling and a set of variable-to-factor messages in BP –*i.e.*, $\underline{\hat{x}} \Leftrightarrow \{\widehat{\mathrm{p}}_{i\to\mathrm{I}}(.)\}_{i,I\in\partial i}$, by defining all the messages leaving variable x_i as a delta-function

$$\widehat{\mathrm{p}}_{i\to\mathrm{I}}(x_i) \quad \stackrel{\text{def}}{=} \quad 1(x_i = \hat{x}_i) \quad \stackrel{\text{def}}{=} \quad \mathrm{G}_{i\to\mathrm{I}}(\underline{\widehat{\mathrm{p}}}_{\Delta i\to\partial i})(x_i) \quad \forall \mathrm{I} \in \partial i \tag{2.79}$$

where $\mathrm{G}_{i\to\mathrm{I}}(\widehat{\mathrm{p}}_{\Delta i\to\partial i})$ is the Gibbs sampling operator that defines variable-to-factor message $\widehat{\mathrm{p}}_{i\to\mathrm{I}}$ as a function of all the messages from Markov blanket of i (Δi) to its adjacent factors (∂i). To completely define this random operator, note that $\hat{x}_i$ is a sample from the conditional distribution of Gibbs sampling

$$\begin{aligned} \hat{x}_i \sim \widehat{\mathrm{p}}(x_i) \quad &\propto \quad \prod_{\mathrm{J}\in\partial i} \mathrm{f}_{\mathrm{I}}(x_i, \underline{\hat{x}}_{\partial \mathrm{I}\setminus i}) \\ &\propto \quad \prod_{\mathrm{I}\in\partial i} \sum_{x_{\setminus i}} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \prod_{j\in\partial \mathrm{I}\setminus i} \widehat{\mathrm{p}}_{j\to\mathrm{I}}(x_j) \end{aligned} \tag{2.80}$$

Combining the operators

Now, lets write the BP updates for sum-product semiring once again; by substituting the factor-to-variable messages (equation (2.6)) into variable-to-factor messages and the marginals (equations (2.7) and (2.9)) we get

$$\widehat{\mathrm{p}}_{i\to \mathrm{I}}(x_i) \quad \propto \quad \prod_{\mathrm{J}\in\partial i\setminus \mathrm{I}} \sum_{X_{\partial \mathrm{J}\setminus i}} \mathrm{f}_{\mathrm{J}}(\underline{x}_{\mathrm{J}}) \prod_{j\in\partial \mathrm{J}\setminus i} \widehat{\mathrm{p}}_{j\to \mathrm{J}}(x_j) \quad \propto \quad \mathrm{P}_{i\to \mathrm{I}}(\widehat{\underline{\mathrm{p}}}_{\underline{\Delta} i\to \partial i})(x_i) \tag{2.81}$$

$$\widehat{\mathrm{p}}(x_i) \quad \propto \quad \prod_{\mathrm{I}\in\partial i} \sum_{X_{\partial \mathrm{I}\setminus i}} \mathrm{f}_{\mathrm{I}}(\underline{x}_I) \prod_{j\in\partial \mathrm{I}\setminus i} \widehat{\mathrm{p}}_{j\to \mathrm{I}}(x_j) \tag{2.82}$$

where, similar to equation (2.7), $\mathrm{P}_{i\to \mathrm{I}}(.)$ denotes the message update operator, with the distinction that here, the arguments are also variable-to-factor messages (rather than factor-to-variable messages).

By this rewriting of BP updates, the BP marginals equation (2.82) are identical in form to the Gibbs sampling distribution of equation (2.80). This similar form allows us to combine the operators linearly to get perturbed BP operator:

$$\mathrm{X}_{i\to \mathrm{I}}(\widehat{\underline{\mathrm{p}}}_{\underline{\Delta} i\to \partial i}) \stackrel{\text{def}}{=} \gamma\ \mathrm{G}_{i\to \mathrm{I}}(\widehat{\underline{\mathrm{p}}}_{\underline{\Delta} i\to \partial i}) + (1-\gamma)\mathrm{P}_{i\to \mathrm{I}}(\widehat{\mathrm{p}}_{\Delta i\to \partial i}) \quad \forall i, \mathrm{I}\in\partial i \tag{2.83}$$

The Perturbed BP operator $\mathrm{X}_{i\to \mathrm{I}}(.)$ updates each message by calculating the outgoing message according to BP and GS operators and linearly combines them to get the final massage. During T iterations of Perturbed BP, the parameter γ is gradually and linearly changed from 0 towards 1.[10] Algorithm 2 summarizes this procedure. Note that the updates of perturbed BP are compatible with variable synchronous message update (see section 2.1.1).

Experiments

Perturbed BP is most successful in solving CSPs; see chapter 3. However we can also use it for marginalization by sampling (equation (2.74)). Here we use the spin-glass Ising model on 8×8 grids and Erdős-Rény (ER) random graphs with $N = 50$ and 150 edges. We sampled local fields independently from $\mathcal{N}(0,1)$ and interactions from $\mathcal{N}(0,\theta^2)$, where we change θ to control the problem difficulty – higher values correspond to more difficult inference problems. We then compared the average of the logarithm (base 10) of mean (over N variables) marginal error

$$\log(\frac{1}{N}\sum_{i,x_i} |\mathrm{p}(x_i) - \widehat{\mathrm{p}}(x_i)|) \tag{2.85}$$

Figure 2.7 compares perturbed BP, Gibbs sampling and BP where larger circles correspond to

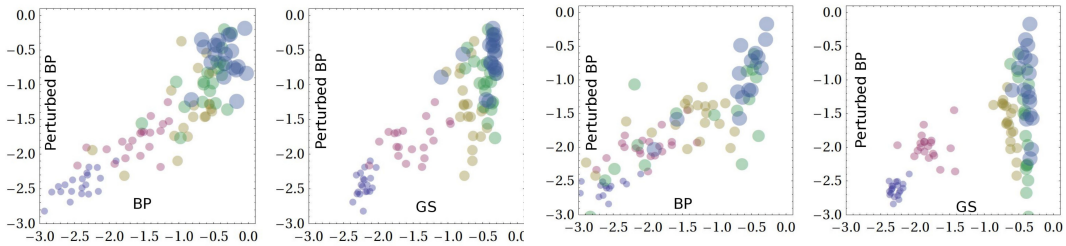
[10] Perturbed BP updates can also be used with any fixed $\gamma \in [0,1]$.

input : a factor graph, number of iterations T.
output: a sample $\underline{\hat{x}}$.
Initialize messages
$\gamma \leftarrow 0$
repeat
  for ***each*** *variable* x_i **do**
    calculate $\widehat{\mathrm{p}}(x_i)$ using equation (2.80)
    calculate BP messages $\widehat{\mathrm{p}}_{i\to I}(.)$ using equation (2.81) $\quad \forall \mathrm{I} \in \partial i$
    sample $\hat{x}_i \sim \widehat{\mathrm{p}}(x_i)$
    combine BP and Gibbs sampling messages:

$$\widehat{\mathrm{p}}_{i\to \mathrm{I}}(x_i) \;\leftarrow\; \gamma\, \widehat{\mathrm{p}}_{i\to \mathrm{I}}(x_i) \;+\; (1-\gamma)\; 1(x_i = \hat{x}_i) \tag{2.84}$$

  end
  $\gamma \leftarrow \gamma + \frac{1}{T-1}$
until *T iterations*
return $\underline{\hat{x}}$

Algorithm 2: Perturbed Belief Propagation



(a) *Ising grid* ***(b)*** *Random ER graph*

Figure 2.7: *Log mean marginal error (x and y axes) comparison between Perturbed BP, BP and Gibbs sampling for* ***(left)*** *8x8 periodic Ising grid;* ***(right)*** *random graph with 50 variables, 150 edges and spin-glass interactions. The size of each circle is proportional to the difficulty of that problem.*

more difficult instances $\theta \in \{0.5, 1, 2, 4, 8\}$. All methods are given a maximum of 10,000 iterations. Perturbed BP and Gibbs sampling use $T = 100$ iterations to obtain each sample, while BP is ran once until convergence or until the maximum number of iterations is reached.

The results shows that Perturbed BP as a sampling method is generally better than Gibbs Sampling. Also, for some cases in which BP's result is very close to random (*i.e.*, $\sim -.3$ in log marginal error), Perturbed BP produces relatively better results. Note that for difficult instances, increasing T even by 100 folds does not significantly improve the results for either Gibbs sampling or perturbed BP. For Gibbs sampling, this can be explained by formation of pure states that result in exponential mixing time [192, 208].

2.6.4 Perturbed survey propagation

When the SP operators are sum-product – *i.e.*, $\widehat{\oplus}$ = sum and $\widehat{\otimes}$ = prod – we can apply a perturbation scheme similar to perturbed BP.

Recall that sum-product SP defines a distribution over BP fixed point. Therefore sampling from this distribution amounts to randomly selecting a single BP fixed point. This corresponds to sampling a single message $\widehat{\mathsf{p}}_{i\to\mathrm{I}}[1] \sim \mathsf{S}_{i\to\mathrm{I}}(\widehat{\mathsf{p}}_{i\to\mathrm{I}})$[11] and bias the SP message $\mathsf{S}_{i\to\mathrm{I}}(.)$ towards this random choice – *i.e.*, (analogous to equation (2.84) in algorithm 2)

$$\begin{aligned} \mathsf{S}_{i\to\mathrm{I}}(\widehat{\mathsf{p}}_{i\to\mathrm{I}}) \quad &\leftarrow \quad \gamma\ \mathsf{S}_{i\to\mathrm{I}}(\widehat{\mathsf{p}}_{i\to\mathrm{I}}) \ +\ (1-\gamma)\mathbb{1}(\widehat{\mathsf{p}}_{i\to\mathrm{I}} = \widehat{\mathsf{p}}_{i\to\mathrm{I}}[1]) \\ \text{where} \quad &\widehat{\mathsf{p}}_{i\to\mathrm{I}}[1] \sim \mathsf{S}_{i\to\mathrm{I}}(\widehat{\mathsf{p}}_{i\to\mathrm{I}}) \end{aligned} \tag{2.86}$$

An alternative form of perturbation is to perturb SP messages using implicit SP marginals. Recall that in using counting SP, the SP marginals over BP marginals ($\mathsf{S}(\widehat{\mathsf{p}})(x_i)$; see equation (2.70)) are simply the frequency of observing a particular marginal in BP fixed points. This implicitly defines SP marginal over the original domains $\mathcal{X}_i; \forall i$, which we denote by $\mathsf{S}(x_i)$

$$\mathsf{S}(x_i) \propto \sum_{\widehat{\mathsf{p}}} \mathsf{S}(\widehat{\mathsf{p}})(x_i) \tag{2.87}$$

After obtaining a sample $\hat{x}_i \sim \mathsf{S}(x_i)$, we bias all the outgoing SP messages accordingly

$$\begin{aligned} \mathsf{S}_{i\to\mathrm{I}}(\widehat{\mathsf{p}}_{i\to\mathrm{I}}) \quad &\leftarrow \quad \gamma\ \mathsf{S}_{i\to\mathrm{I}}(\widehat{\mathsf{p}}_{i\to\mathrm{I}}) \ +\ (1-\gamma)\ \mathbb{1}\Big(\widehat{\mathsf{p}}_{i\to\mathrm{I}}(.) = \mathbb{1}(\hat{x}_i, .)\Big) \quad \forall \mathrm{I} \in \partial i \\ \text{where} \quad &\hat{x}_i \sim \mathsf{S}_i(x_i) \end{aligned} \tag{2.88}$$

where, similar to perturbed BP, γ is gradually increased from 0 to 1 during T iterations of Perturbed SP. We use this form of perturbation in section 3.3 to obtain a satisfying assignment $\underline{\hat{x}}$, to CSPs. We show that although computationally more expensive than perturbed BP, this method often outperforms all the other well-known methods in solving random CSPs.

[11] Recall that the SP marginal over each message $\widehat{\mathsf{p}}_{i\to\mathrm{I}}$ is identical to the corresponding message $\mathsf{S}_{i\to\mathrm{I}}(\widehat{\mathsf{p}}_{i\to\mathrm{I}})$.

Part II

Combinatorial problems

Message Passing algorithms of different semirings are able to solve a variety of combinatorial problems: (I) To solve **constraint satisfaction problems (CSPs)** the sum-product message passing is often used, where $p(\underline{x})$ defines a uniform distribution over solutions and the objective is to produce a single assignment $\underline{x}^*$ s.t. $p(\underline{x}^*) > 0$. (II) Here the estimates of the partition function, either using the approximation given by the Bethe free energy (section 2.3.1) or the decomposition of integral in section 2.5.1, is used for **approximate counting** of the number of solutions. This estimate to the partition function is also used for integration problems such approximating the permanent of a matrix (see chapter 5). (III) The min-sum semiring is often used for **(constrained) optimization** and we formulate (IV) **bottleneck problems** as min-max inference.

This part of the thesis studies the message passing solutions to combinatorial problems under three broad categories. (1) Chapter 3 studies the constraint satisfaction problems, where we use perturbed message passing (section 2.6.3 and 2.6.4) to produce state-of-the-art results in solving random instances of satisfiability and coloring problems in section 3.3. This chapter then studies several other $\mathbb{NP}$-hard problems including set-cover, independent set, max-clique, clique-cover and packing for construction of non-linear codes. (2) Chapter 4 studies variations of clustering problems including k-median, k-center, k-clustering, hierarchical clustering and modularity optimization. (3) In chapter 5 we study problems that involve enumeration, constraint satisfaction or constrained optimization over permutations. This includes (bottleneck) travelling salesman problem, matching, graph alignment, graph isomorphism and finding symmetries.

Note that this classification of combinatorial problems into three categories is superficial and is made solely to provide some organization. In several places we violate this categorization in favour of better flow. For example we study some constraint satisfaction problems such as (sub)-graph isomorphism and Hamiltonian cycle in chapter 5 rather than chapter 3. We investigate the "optimization" counterpart of some CSPs in chapter 3 and review message passing solutions to finding trees rather than clusters in chapter 4. Moreover, many of the graphical models presented here are proposed by other researchers and they are included here only for completeness. As a final remark, we note that many of the statements in the following are assuming $\mathbb{P} \neq \mathbb{NP}$.

Chapter 3

Constraint satisfaction

We saw in section 1.3 that "any" semiring can formulate Constraint Satisfaction Problems (CSPs). In particular, as we saw in section 1.3, several semirings are isomorphic to the and-or $(\{\text{FALSE}, \text{TRUE}\}, \vee, \wedge)$ semiring and therefore result in equivalent BP procedures. The BP message update over the and-or semiring is called **warning propagation** (WP). WP marginals indicate whether or not a particular assignment to each variable is allowed, and therefore indicate a *cluster of solutions*. However, the success of warning propagation highly depends on initialization of messages. In contrast, if convergent, the fixed points of BP on the sum-product semiring $(\mathbb{R}^{\geq 0}, +, \times)$ are less dependent on initialization.

Example 3.0.1. K**-coloring:** Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, the K-coloring (K-COL) problem asks whether it is possible to assign one color (out of K) to each node s.t. no two adjacent nodes have the same color. Here, $x_i \in \mathcal{X}_i = \{1, \dots, q\}$ is a K-ary variable for each $i \in \mathcal{N}$, and we have $M = |\mathcal{E}|$ constraints; each constraint $\mathsf{f}_{i,j}(x_i, x_j) = 1(x_i \neq x_j)$ depends only on two variables and is satisfied iff the two variables have different values. Here the identity function $1(x_i \neq x_j)$ depends on the semiring (see section 1.3).

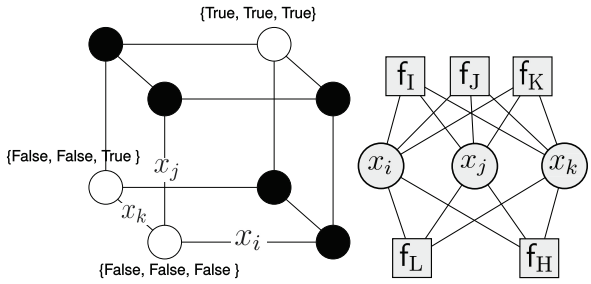


Figure 3.1: *(a) The set of all possible assignments to 3 variables. The solutions to the 3-SAT problem of equation (3.1) are in white circles. (b) The factor-graph corresponding to the 3-SAT problem of equation (3.1). Here each factor prohibits a single assignment.*

Example 3.0.2. *K*-**Satisfiability (K-SAT):** Given a conjunction of disjunctions with K literals, K-satisfiability seeks an assignment that evaluates to TRUE. Here, all variables are binary ($\mathcal{X}_i = \{\text{TRUE}, \text{FALSE}\}$) and each clause (factor f_I) depends on $K = \mid \partial I \mid$ variables. A clause evaluates to zero only for a single assignment out of 2^K possible assignment of variables [107].

Consider the following (or-and semiring formulation of) 3-SAT problem over 3 variables with 5 clauses:

$$q(\underline{x}) = \underbrace{(\neg x_i \vee \neg x_j \vee x_k)}_{f_I} \wedge \underbrace{(\neg x_i \vee x_j \vee x_k)}_{f_J} \wedge \underbrace{(x_i \vee \neg x_j \vee x_k)}_{f_K} \wedge \underbrace{(\neg x_i \vee x_j \vee \neg x_k)}_{f_L} \wedge \underbrace{(x_i \vee \neg x_j \vee \neg x_k)}_{f_H} \tag{3.1}$$

The factor f_I corresponding to the first clause takes the value $\overset{\otimes}{1}$ (for or-and semiring this corresponds to $\overset{\wedge}{1} = \text{TRUE}$), except for $\underline{x}_I = (\text{TRUE}, \text{TRUE}, \text{FALSE})$, in which case it is equal to $\overset{\oplus}{1}$ ($\overset{\vee}{1} = \text{FALSE}$). Figure 3.1 shows this factor-graph and its set of solutions:
$\mathcal{S} = \Big\{(\text{TRUE}, \text{TRUE}, \text{TRUE}), (\text{FALSE}, \text{FALSE}, \text{FALSE}), (\text{FALSE}, \text{FALSE}, \text{TRUE})\Big\}$.

When using sum-product semiring, where $f_I(\underline{x}) \in \mathcal{Y}_I = \{0, 1\}$, $p(\underline{x})$ (equation (1.17)) defines a uniform distribution over the set of solutions and the partition function $q(\emptyset)$ counts the number of solutions. The challenge is then to sample from this distribution (or estimate $q(\emptyset)$). The common approach to sample from $p(\underline{x})$ is to use decimation (see section 1.4.1). Here one repeatedly applies sum-product BP to estimate marginals $\widehat{p}(x_i)$. Then one fixes a subset of variables $\underline{x}_A$ according to their marginals. For this one may sample $x_i^* \sim \widehat{p}(x_i)$ or select x_i with maximum marginal $x_i^* = \arg_{x_i} \max \widehat{p}(x_i)$. Sum-product BP is then applied to the reduced factor-graph in which $f_I(\underline{x}_I)$ is replaced by $f_I(\underline{x}_I)1(\underline{x}_{I\cap A} = \underline{x}^*_{I\cap A})$. This process, called **sum-product BP-guided-decimation** (BP-dec), is repeated to obtain a complete joint assignment $\underline{x}^*$.

However, using BP-guided-decimation is solving a more difficult problem of marginalization. In fact, in section 1.4.1 we showed how using decimation one may estimate the partition function (which is a $\#\mathbb{P}$ problem). This suggests that decimation may not be the most efficient approach to solving $\mathbb{NP}$-complete CSPs. Here, instead we consider using the perturbed belief propagation (section 2.6.3) to sample from the set of solutions, where the semiring used by perturbed BP is the same as sum-product BP-dec.

To better understand warning propagation, sum-product BP-dec, and perturbed BP, when applied to CSPs, consider the following examples.

Example 3.0.3. Here we apply three different message passing methods to solve the simple 3-SAT example of figure 3.1.

(I) Warning Propagation:
We use the max-product semiring $(\{0,1\}, \max, \text{prod})$ version of warning propagation for this example. As figure 3.1 suggests, the set of solutions $\mathcal{S}$ clusters into two subsets $\{\{\text{TRUE}, \text{TRUE}, \text{TRUE}\}\}$ and $\{\{\text{FALSE}, \text{FALSE}, \text{FALSE}\}, \{\text{FALSE}, \text{FALSE}, \text{TRUE}\}\}$. Here, each of the clusters is a fixed point for WP – *e.g.*, the cluster with two solutions corresponds to the following fixed point

$$\begin{aligned}
&\widehat{\mathrm{p}}_{i\to \mathrm{A}}(\text{TRUE}) = \widehat{\mathrm{p}}(x_i = \text{TRUE}) = 0 && \\
&\widehat{\mathrm{p}}_{i\to \mathrm{A}}(\text{FALSE}) = \widehat{\mathrm{p}}(x_i = \text{FALSE}) = 1 && \forall \mathrm{A} \in \partial i \\
&\widehat{\mathrm{p}}_{j\to \mathrm{A}}(\text{TRUE}) = \widehat{\mathrm{p}}(x_j = \text{TRUE}) = 0 && \\
&\widehat{\mathrm{p}}_{j\to \mathrm{A}}(\text{FALSE}) = \widehat{\mathrm{p}}(x_j = \text{FALSE}) = 1 && \forall \mathrm{A} \in \partial j \\
&\widehat{\mathrm{p}}_{k\to \mathrm{A}}(\text{TRUE}) = \widehat{\mathrm{p}}(x_k = \text{TRUE}) = 1 && \\
&\widehat{\mathrm{p}}_{k\to \mathrm{A}}(\text{FALSE}) = \widehat{\mathrm{p}}(x_k = \text{FALSE}) = 1 && \forall \mathrm{A} \in \partial k
\end{aligned}$$

where the messages indicate the allowed assignments within this particular cluster of solutions. Depending on the initialization, WP messages may converge to any of its fixed points that also include the trivial cluster, where all (alternatively none) of the assignments are allowed.

(II) BP-dec:
Applying BP to this 3-SAT problem (starting from uniform messages) takes 20 iterations to converge – *i.e.*, for the maximum change in the marginals to be below $\epsilon = 10^{-9}$. Here the message, $\widehat{\mathrm{p}}_{\mathrm{I}\to i}(x_i)$, from f_I to x_i is:

$$\widehat{\mathrm{p}}_{\mathrm{I}\to 1}(x_i) \quad \propto \quad \sum_{\underline{x}_{j,k}} \mathrm{f}_\mathrm{I}(\underline{x}_{1,2,3})\ \widehat{\mathrm{p}}_{j\to \mathrm{I}}(x_j)\ \widehat{\mathrm{p}}_{k\to \mathrm{I}}(x_k)$$

Similarly, the message in the opposite direction, $\widehat{\mathrm{p}}_{k\to \mathrm{I}}(x_i)$ is

$$\widehat{\mathrm{p}}_{i\to \mathrm{I}}(x_i) \quad \propto \quad \widehat{\mathrm{p}}_{\mathrm{J}\to i}(x_i)\ \widehat{\mathrm{p}}_{\mathrm{K}\to i}(x_i)\ \widehat{\mathrm{p}}_{\mathrm{L}\to i}(x_i)\ \widehat{\mathrm{p}}_{\mathrm{H}\to i}(x_i)$$

Here BP gives us the following approximate marginals: $\widehat{\mathrm{p}}(x_i = \text{TRUE}) = \widehat{\mathrm{p}}(x_j = \text{TRUE}) = .319$ and $\widehat{\mathrm{p}}(x_k = \text{TRUE}) = .522$. From the set of solutions, we know that the correct marginals are $\mathrm{p}(x_i = \text{TRUE}) = \mathrm{p}(x_j = \text{TRUE}) = 1/3$ and $\mathrm{p}(x_k = \text{TRUE}) = 2/3$. The error of BP is caused by influential loops in the factor-graph of figure 3.1(b). Here the error is rather small; it can be arbitrarily large in some instances; sometimes it ca prevent converging at all.

By fixing the value of x_i to FALSE, the SAT problem of equation (3.1) collapses to:

$$SAT(\underline{x}_{\{j,k\}} \mid x_i = \text{FALSE}) \ = \ (\neg x_j \vee x_k) \wedge (\neg x_j \vee \neg x_k) \tag{3.2}$$

BP-dec applies BP again to this reduced problem, which give $\widehat{\mathrm{p}}(x_j = \textsc{true}) = .14$ (note here that $\mathrm{p}(x_j = \textsc{true}) = 0$) and $\widehat{\mathrm{p}}(x_k = \textsc{true}) = 1/2$. By fixing x_j to FALSE, another round of decimation yields a solution $\underline{x}^* = \{\textsc{false}, \textsc{false}, \textsc{true}\}$.

(III) Perturbed Belief Propagation:
Perturbed BP can find a solution in $T = 4$ iterations (see algorithm 2 on page 71). Our implementation shuffles the order of updates for variables in each iteration.

In the first iteration, $\gamma = 0$, which means updates are the same as that of sum-product BP. In the second iteration, the order of updates is x_j, x_k, x_i and $\gamma = 1/3$. At the end of this iteration $\widehat{\mathrm{p}}^{(\mathrm{t}=2)}(x_j = \textsc{true}) = .38$. Perturbed BP then samples $\hat{x}_j = \textsc{false}$ from this marginal. This sample influences the outgoing message according to the perturbed BP update equation (2.84), which in turn influences the beliefs for x_i and x_k. At the end of this iteration $\widehat{\mathrm{p}}^{(\mathrm{t}=2)}(x_i = \textsc{true}) = .20$ and $\widehat{\mathrm{p}}^{(\mathrm{t}=2)}(x_k = \textsc{true}) = .53$. At the final iteration $\gamma = 1$ and the order of updates is x_i, x_j and x_k. At this point $\widehat{\mathrm{p}}^{(\mathrm{t}=3)}(x_i = \textsc{true}) = .07$ and the sample $\hat{x}_i = \textsc{false}$. This means the outgoing message is deterministic (*i.e.*, $\widehat{\mathrm{p}}_{i\to\mathrm{A}}(\textsc{false}) = 1$ and $\widehat{\mathrm{p}}_{i\to\mathrm{A}}(\textsc{true}) = 0$, for all $\mathrm{A} \in \partial i$). This choice propagates to select $\hat{x}_j = \textsc{false}$. Finally $\widehat{\mathrm{p}}^{(\mathrm{t}=3)}(x_k = \textsc{true}) = \widehat{\mathrm{p}}^{(\mathrm{t}=3)}(x_k = \textsc{true}) = .5$, which correctly shows that both choices for $\hat{x}_k$ produce a solution.

To compare the performance of sum-product BP-dec and Perturbed BP on general CSPs, we considered all CSP instances from XCSP repository [189, 270], that do not include global constraints or complex domains. All instances with intensive constraints (*i.e.*, functional form) were converted into extensive format for explicit representation using dense factors. We further removed instances containing constraints with more that 10^6 enteries in their tabular form. We also discarded instances that collectively had more than 10^8 enteries in the dense tabular form of their constraints.[1]

Figure 3.2(a,b) compares the time and iterations of BP-dec and Perturbed BP for successful attempts where both methods satisfied an instance. [2]

Overall Perturbed BP, with 284 solved instances, is more successful than BP-dec with 253 successful runs. On the other hand, the average number of iterations for successful instances of BP-dec is 41,284, compared to 133 iterations for Perturbed BP. This makes Perturbed BP 300 times more

[1] Since our implementation represents all factors in a dense tabular form, we had to remove many instances because of their large factor size. We anticipate that Perturbed BP and BP-dec could probably solve many of these instances using a sparse representation of factors.

[2] We used a convergence threshold of $\epsilon = .001$ for BP and terminated if the threshold was not reached after $T = 10 \times 2^{10} = 10,240$ iterations. To perform decimation, we sort the variables according to their bias and fix ρ fraction of the most biased variables in each iteration of decimation. This fraction, ρ, was initially set to 100%, and it was divided by 2 each time BP-dec failed on the same instance. BP-dec was repeatedly applied using the reduced ρ, at most 10 times, unless a solution was reached – *i.e.*, $\rho = .1\%$ at final attempt.

For Perturbed BP, $T = 10$ at the starting attempt, which was increased by a factor of 2 in case of failure. This was repeated at most 10 times which means Perturbed BP used $T = 10,240$ at its final attempt. Note that Perturbed BP at most uses the same number of iterations as the maximum iterations per single iteration of decimation in BP-dec.

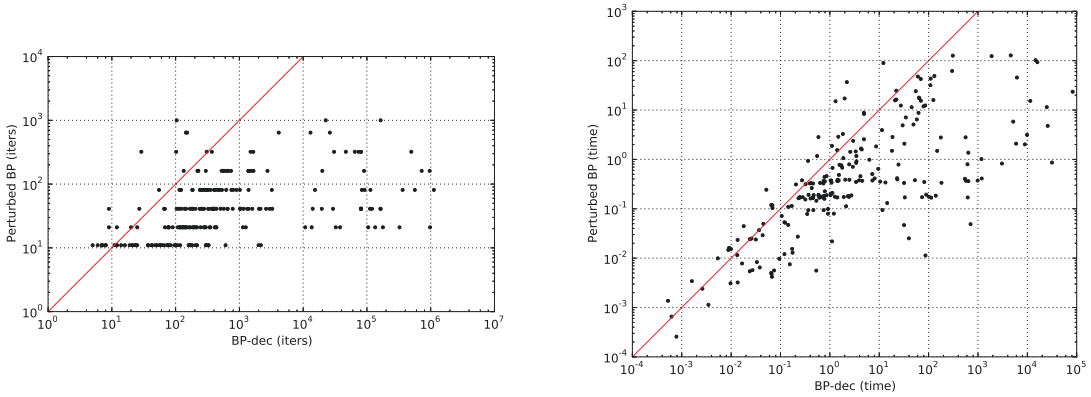


Figure 3.2: *Comparison of number of iterations (left) and time (right) used by BP-dec and Perturbed BP in benchmark instances where both methods found satisfying assignments.*

efficient than BP-dec. [3]

3.1 Phase transitions in random CSPs

Random CSP (rCSP) instances have been extensively used in order to study the properties of combinatorial problems [1, 103, 179, 215] as well as in analysis and design of algorithms [213, 277].

Studies of rCSP, as a critical phenomena, focus on the geometry of the solution space as a function of the problem's difficulty, where rigorous [2, 70] and non-rigorous [209, 210] analyses have confirmed the same geometric picture.

When working with large random instances, a scalar α associated with a problem instance, a.k.a. **control parameter** – *e.g.*, the clause to variable ratio in SAT– can characterize that instance's difficulty (*i.e.*, larger control parameter corresponds to a more difficult instance) and in many situations it characterizes a sharp transition from satisfiability to unsatisfiability [59].

Example 3.1.1. Random K-satisfiability Random K-SAT instance with N variables and $M = \alpha N$ constraints are generated by selecting K variables at random for each constraint. Each constraint is set to zero (*i.e.*, unsatisfied) for a single random assignment (out of 2^K). Here α is the control parameter.

Example 3.1.2. Random K-coloring The control parameter for a random K-COL instances with N variables and M constraints is its average degree $\alpha = \frac{2M}{N}$. We consider Erdős-Rény random

[3] We also ran BP-dec on all the benchmarks with maximum number of iterations set to $T = 1000$ and $T = 100$ iterations. This reduced the number of satisfied instances to 249 for $T = 1000$ and 247 for $T = 100$, but also reduced the average number of iterations to 1570 and 562 respectively, which are still several folds more expensive than Perturbed BP. see Appendix xyz for more details on these results.

graphs and generate a random instance by sequentially selecting two distinct variables out of N at random to generate each of M edges. For large N, this is equivalent to selecting each possible factor with a fixed probability, which means the nodes have Poisson degree distribution $\Pr(|\partial i| = d) \propto e^{-\alpha} \alpha^{d}$.

While there are tight bounds for some problems [3], finding the exact location of this transition for different CSPs is still an open problem. Besides transition to unsatisfiability, these analyses have revealed several other (phase) transitions [179]. Figure 3.3(a)-(c) shows how the geometry of the set of solutions changes by increasing the control parameter.

Here we enumerate various phases of the problem for increasing values of the control parameter: **(a)** In the so-called **Replica Symmetric** (RS) phase, the symmetries of the set of solutions (a.k.a. ground states) reflect the trivial symmetries of problem wrt variable domains. For example, for K-COL the set of solutions is symmetric wrt swapping all red and blue assignment. In this regime, the set of solutions form a **giant cluster** (*i.e.*, a set of neighboring solutions), where two solutions are considered neighbors when their Hamming distance is one [2] (or non-divergent with number of variables [210]. Local search methods (*e.g.*, [277]) and BP-dec can often efficiently solve random CSPs that belong to this phase.

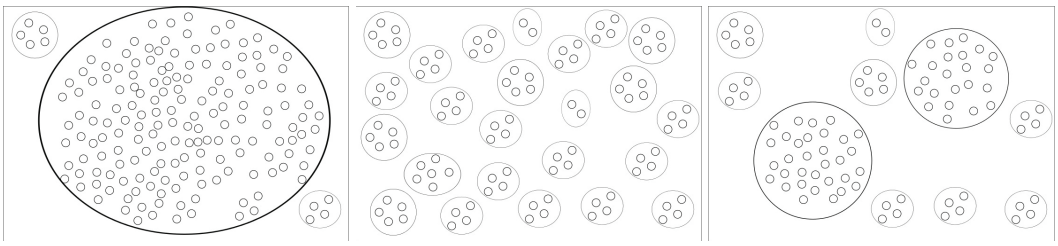


Figure 3.3: *A 2-dimensional schematic view of how the set of solutions of CSP varies as we increase the control parameter α from (left) replica symmetric phase to (middle) clustering phase to (right) condensation phase. Here small circles represent solutions and the bigger circles represent clusters of solutions. Note that this view is very simplistic in many ways* – e.g., *the total number of solutions and the size of clusters should generally decrease from left to right.*

(b) In **clustering** or **dynamical** transition (1dRSB[4]), the set of solutions decomposes into an exponential number of distant clusters. Here two clusters are distant if the Hamming distance between their respective members is divergent (*e.g.*, linear) in the number of variables. **(c)** In the **condensation** phase transition (1sRSB[5]), the set of solutions condenses into a few dominant clusters. Dominant clusters have roughly the same number of solutions and they collectively contain almost all of the solutions. While SP can be used even within the condensation phase, BP usually

[4] 1st order dynamical RSB. Symmetry Breaking is a general term indicating a phenomenon during which a system is breaking the symmetry that governs its behaviour by selecting a particular branch. The term Replica Symmetry Breaking (RSB) originates from the technique –*i.e.*, Replica trick ([211])– that was first used to analyze this setting. According to RSB, the trivial symmetries of the problem do not characterize the clusters of solution.

[5] 1st order static RSB.

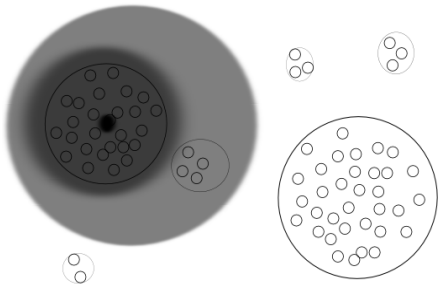


Figure 3.4: *This schematic view demonstrates the clustering during condensation phase. Here assume x and y axes correspond to x_i and x_j. Considering the whole space of assignments, x_i and x_j are highly correlated. The formation of this correlation between distant variables on a factor-graph breaks BP. Now assume that Perturbed BP messages are focused on the largest shaded ellipse. In this case the correlation is significantly reduced.*

fails to converge in this regime. However each cluster of solutions in the clustering and condensation phase is a valid fixed-point of BP. **(d)** A **rigidity** transition (not included in figure 3.3) identifies a phase in which a finite portion of variables are fixed within dominant clusters. This transition triggers an exponential decrease in the total number of solutions, which leads to **(e)** unsatisfiability transition.[6] This rough picture summarizes first order Replica Symmetry Breaking's (1RSB) basic assumptions [208].

3.1.1 Pitfalls of decimation

Previously we gave an argument against decimation, based on the complexity of marginalization and integration. Some recent analyses draw similarly negative conclusions on the effect of decimation [71, 218, 261]. The general picture is that at some point during the decimation process, variables form long-range correlations such that fixing one variable may imply an assignment for a portion of variables that form a loop, potentially leading to contradictions. Alternatively the same long-range correlations result in BP's lack of convergence and error in marginals that may lead to unsatisfying assignments.

Perturbed BP avoids the pitfalls of BP-dec in two ways: (I) Since many configurations have non-zero probability until the final iteration, perturbed BP can avoid contradictions by adapting to the most recent choices. This is in contrast to decimation in which variables are fixed once and are unable to change afterwards. Some backtracking schemes [240] attempt to fix this problem with decimation. (II) We speculate that simultaneous bias of all messages towards sub-regions, prevents the formation of long-range correlations between variables that breaks BP in 1sRSB; see figure 3.4.

[6] In some problems, the rigidity transition occurs before condensation transition.

3.2 Revisiting survey propagation

SP is studied on random (hyper) graphs representing CSPs at thermodynamic limit (*i.e.*, as $N \to \infty$). Large random graphs are locally tree-like, which means the length of short loops are typically in the order of $\log(N)$ [145]. This ensures that, in the absence of long-range correlations, BP is asymptotically exact, as the set of messages incoming to each node or factor are almost independent. Although BP messages remain uncorrelated until the condensation transition [179], the BP equations do not completely characterize the set of solutions after the clustering transition. This inadequacy is indicated by the existence of a set of several valid fixed points (rather than a unique fixed-point) for WP as an instance of BP. For a better intuition, consider the cartoons of figure 3.3(middle) and (right). During the clustering phase (middle), x_i and x_j (corresponding to the x and y axes) are not highly correlated, but they become correlated during and after condensation (right). This correlation between variables that are far apart in the factor-graph results in correlation between BP messages. This is because it implies that even if loops are long, they remain influential. This violates BP's assumption that messages are uncorrelated, which results in BP's failure in this regime.

This is where survey propagation comes into the picture in solving CSPs. Going back to our algebraic notation for SP, using counting SP with warning propagation semiring $(\{0,1\}, \max, \text{prod})$ as the initial semiring and sum-product $(\mathbb{R}^{\geq 0}, \text{sum}, \text{prod})$ as the SP semiring, is computationally tractable. This is because $\mathcal{Y}^* = \{0,1\}$ in the initial semiring is finite, and therefore each message can have finite number of $2^{|\mathcal{X}_i|}$ values

$$\widehat{\mathsf{p}}_{i\to \mathrm{I}}(x_i), \widehat{\mathsf{p}}_{\mathrm{I}\to i}(.) \in \Big\{(0,\dots,0),(0,\dots,0,1),(0,\dots,1,0),\dots,(1,\dots,0),\dots,(1,\dots,1)\Big\} \quad \forall i, \mathrm{I} \in \partial i$$

This means each SP message is a distribution over these possibilities $\mathsf{S}_{i\to \mathrm{I}}(\widehat{\mathsf{p}}_{i\to \mathrm{I}}) \in \mathbb{R}^{2^{|\mathcal{X}_i|}}$. However since $(0,\dots,0)$ indicates an unfeasible case, where no assignment is allowed, we explicitly ignore it in SP message updates. This gives us the following update equations and marginals for SP when applied to CSPs

$$\mathsf{S}_{i\to \mathrm{I}}(\widehat{\mathsf{p}}_{i\to \mathrm{I}}) \propto \sum_{\underline{\widehat{\mathsf{p}}}_{\partial i\setminus \mathrm{I}\to i}} 1\Big(\widehat{\mathsf{p}}_{i\to \mathrm{I}}(.) = \prod_{\mathrm{J}\in\partial i\setminus \mathrm{I}} \widehat{\mathsf{p}}_{\mathrm{J}\to i}(.)\Big)\Big(\prod_{\mathrm{J}\in\partial i\setminus \mathrm{I}} \mathsf{S}_{\mathrm{J}\to i}(\widehat{\mathsf{p}}_{\mathrm{J}\to i})\Big) \quad \forall i, \mathrm{I} \in \partial i \tag{3.3}$$

$$\mathsf{S}_{\mathrm{I}\to i}(\widehat{\mathsf{p}}_{\mathrm{I}\to i}) \propto \sum_{\underline{\widehat{\mathsf{p}}}_{\partial \mathrm{I}\setminus i\to \mathrm{I}}} 1\Big(\widehat{\mathsf{p}}_{\mathrm{I}\to i}(.) = \sum_{x_{\setminus i}} \mathsf{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}}) \prod_{j\in\partial \mathrm{I}\setminus i} \widehat{\mathsf{p}}_{j\to \mathrm{I}}(.)\Big)\Big(\prod_{j\in\partial \mathrm{I}\setminus i} \mathsf{S}_{j\to \mathrm{I}}(\widehat{\mathsf{p}}_{j\to \mathrm{I}})\Big) \tag{3.4}$$

$$\mathsf{S}(\widehat{\mathsf{p}}_i) = \sum_{\underline{\widehat{\mathsf{p}}}_{\partial i\to i}} \widehat{1}(\widehat{\mathsf{p}}_i(.) = \prod_{\mathrm{I}\in\partial i} \widehat{\mathsf{p}}_{\mathrm{I}\to i}(.)) \prod_{\mathrm{I}\in\partial i} \mathsf{S}_{\mathrm{I}\to i}(\widehat{\mathsf{p}}_{\mathrm{I}\to i}) \tag{3.5}$$

$$\mathsf{S}_{\mathrm{I}\to i}((0,\dots,0)) = 0 \quad \text{and} \quad \mathsf{S}_{i\to \mathrm{I}}((0,\dots,0)) = 0$$

Example 3.2.1. Consider the SP message $\mathsf{S}_{i\to \mathrm{I}}(\widehat{\mathsf{p}}_{i\to \mathrm{I}})$ in factor-graph of figure 3.1 on page 75. Here the summation in equation (3.3) is over all possible combinations of max-product BP messages

$\widehat{\mathrm{p}}_{\mathrm{J}\to i}\widehat{\mathrm{p}}_{\mathrm{K}\to i},\widehat{\mathrm{p}}_{\mathrm{L}\to i},\widehat{\mathrm{p}}_{\mathrm{H}\to i}$. Since each of these messages can assume one of the three valid values – *e.g.*, $\widehat{\mathrm{p}}_{\mathrm{J}\to i}(x_i) \in \{(0,1),(1,0),(1,1)\}$ – for each particular assignment of $\widehat{\mathrm{p}}_{i\to\mathrm{I}}$, a total of 3^4 possible combinations are enumerated in the summations of equation (3.3). However only the combinations that form a valid max-product message update have non-zero contribution in calculating $\mathrm{S}_{i\to\mathrm{I}}$.

3.2.1 Flavours of SP-guided decimation

The SP-marginal over WP marginals (equation (3.5)) also implies a distribution $\mathrm{S}(x_i)$ over the original domain (see equation (2.87)). Similar to BP-dec we can use either the implicit marginals of equation (2.87) or the SP marginals of equation (3.5) to perform decimation. In the former case, which we call SP-dec(S) we select $x_i^* = \arg_{x_i}\max \mathrm{S}(x_i)$ during decimation, and in the later case, which we call SP-dec(C), we clamp $\widehat{\mathrm{p}}_i^* = \arg_{\widehat{\mathrm{p}}_i}\max \mathrm{S}(\widehat{\mathrm{p}}_i)$. This means all the outgoing messages from this variable node in the factor-graph are clamped in the same way – *i.e.*, $\mathrm{S}_{i\to\mathrm{I}}(\widehat{\mathrm{p}}_{i\to\mathrm{I}}) = \widehat{\mathrm{p}}_i^* \quad \forall \mathrm{I} \in \partial i$.

In the first case, SP-dec(S), we expect a single assignment $\underline{x}^*$, while for SP-dec(C) at the end of decimation we should obtain a cluster of solutions, where a subset of assignments is allowed for each x_i. However, during the decimation process (in both SP-dec(S) and SP-dec(C)), usually after fixing a subset of variables, SP marginals, $\mathrm{S}(x_i)$, become close to uniform, indicating that clusters of solution have no preference over particular assignment of the remaining variables. The same happens when we apply SP to random instances in RS phase (figure 3.3(left)). At this point (a.k.a. paramagnetic phase) solutions form a giant cluster and a local search method or BP-dec can often efficiently find an assignment to the variables that are not yet fixed by decimation.

The original decimation procedure for K-SAT [48] corresponds to SP-dec(S). SP-dec(C) for CSP with Boolean variables is only slightly different, as SP-dec(C) can choose to fix a cluster to $\widehat{\mathrm{p}}_i = (1,1)$ in addition to the options of $\widehat{\mathrm{p}}_i = (1,0)$ and $\widehat{\mathrm{p}}_i = (0,1)$ (corresponding to $x_i = 0$ and $x_i = 1$ respectively), available to SP-dec(S). However, for larger domains (*e.g.*, K-COL), SP-dec(C) has a clear advantage. For example, in 3-COL, SP-dec(C) may choose to fix a variable to $\widehat{\mathrm{p}}_{i\to\mathrm{I}} = (0,1,1)$ (*i.e.*, the first color is not allowed) while SP-dec(S) can only choose between $\widehat{\mathrm{p}}_i \in \{(0,0,1),(0,1,0),(1,0,0)\}$. This significant difference is also reflected in their comparative success-rate on K-COL. [7] (See section 3.3)

3.2.2 Computational Complexity

The computational complexity of each SP update of equation (3.4) is $O(2^{|\mathcal{X}_i|}-1)^{|\partial\mathrm{I}|}$ as for each particular value $\widehat{\mathrm{p}}_{i\to\mathrm{I}}$, SP needs to consider every combination of incoming messages, each of which can take $2^{|\mathcal{X}_i|}$ values (minus the empty set). Similarly, using a naive approach the cost of update of equation (3.3) is $O(2^{|\mathcal{X}_i|}-1)^{|\partial i|}$. However by considering incoming messages one at a time, we can perform the same exact update in $O(|\partial i|\ 2^{2|\mathcal{X}_i|})$. In comparison to the cost of BP updates

[7] Previous applications of SP-dec to K-COL by [49] used a heuristic for decimation that is similar SP-dec (C).

(*i.e.*, $O(|\partial i|\ |\mathcal{X}_i|)$ and $O(|\mathcal{X}_{\mathrm{I}}|)$ for two types of message update; see section 2.1), we see that SP updates are substantially more expensive for large domains $|\mathcal{X}_i|$ and higher order factors with large $|\partial \mathrm{I}|$.

3.2.3 Perturbed survey propagation for CSP

Similar to SP, we use perturbed SP with $(\{0,1\}, \max, \mathrm{prod})$ as the first semiring and $(\mathbb{R}, \mathrm{sum}, \mathrm{prod})$ as the second semiring. Since perturbed SP seeks a single assignment, rather than a cluster of solutions, it can find satisfying solutions to paramagnetic instances. This is in contrast to SP-dec, which in paramagnetic cases returns a trivial WP fixed point in which all assignment are allowed. This means as opposed to SP-dec, which is mostly applied to random CSPs in the clustering and condensation phase, perturbed SP can be used to solve non-random and also random instances in RS phase.

To demonstrate this, we applied perturbed SP to benchmark CSP instances of figure 3.2, in which the maximum number of elements in the factor was less than 10.[8] Here perturbed SP solved 80 instances out of 202 cases, in comparison to perturbed BP that solved 78 instances, making perturbed SP slightly better, also in solving real-world problems.

3.3 Satisfiability and coloring

In examples 3.1.1 and 3.1.2, we introduced the random procedures that are often used to produce instances of K-satisfiability and K-coloring problems.

Here we report the results on K-SAT for $K \in \{3,4\}$ and K-COL for $K \in \{3,4,9\}$. We used the procedures to produce 100 random instances with $N = 5,000$ variables for each control parameter α and here report the probability of finding a satisfying assignment for different methods – *i.e.*, the portion of 100 instances that were satisfied by each method.[9]

Figure 3.5(first row) visualizes the success rate of different methods on 3-SAT (right) and 3-COL (left). figure 3.5(second row) reports the number of variables that are fixed by SP-dec(C) and (S) before calling BP-dec as local search. The third row shows the average amount of time that is used to find a satisfying solution. This does not include the failed attempts. For SP-dec variations,

[8] The number of iterations and other settings for perturbed SP were identical to the ones used to compare BP-dec and perturbed BP.

[9] For coloring instances, to help decimation, we break the initial symmetry of the problem by fixing a single variable to an arbitrary value. For BP-dec and SP-dec, we use a convergence threshold of $\epsilon = .001$ and fix $\rho = 1\%$ of variables per iteration of decimation. Perturbed BP and Perturbed SP use $T = 1000$ iterations. Decimation-based methods use a maximum of $T = 1000$ iterations per iteration of decimation. If any of the methods failed to find a solution in the first attempt, T was increased by a factor of 4 at most 3 times – *i.e.*, in the final attempt $T = 64,000$. To avoid blow-up in run-time, for BP-dec and SP-dec, only the maximum iteration, T, during the first iteration of decimation, was increased (this is similar to the setting of [48] for SP-dec). For both variations of SP-dec (see section 3.2.1) after each decimation step, if $\max_{i,x_i} \widehat{\mathrm{p}}(x_i) - \frac{1}{|\mathcal{X}_i|} < .01$ (*i.e.*, marginals are close to uniform) we consider the instance para-magnetic, and run BP-dec (with $T = 1000$, $\epsilon = .001$ and $\rho = 1\%$) on the simplified instance.

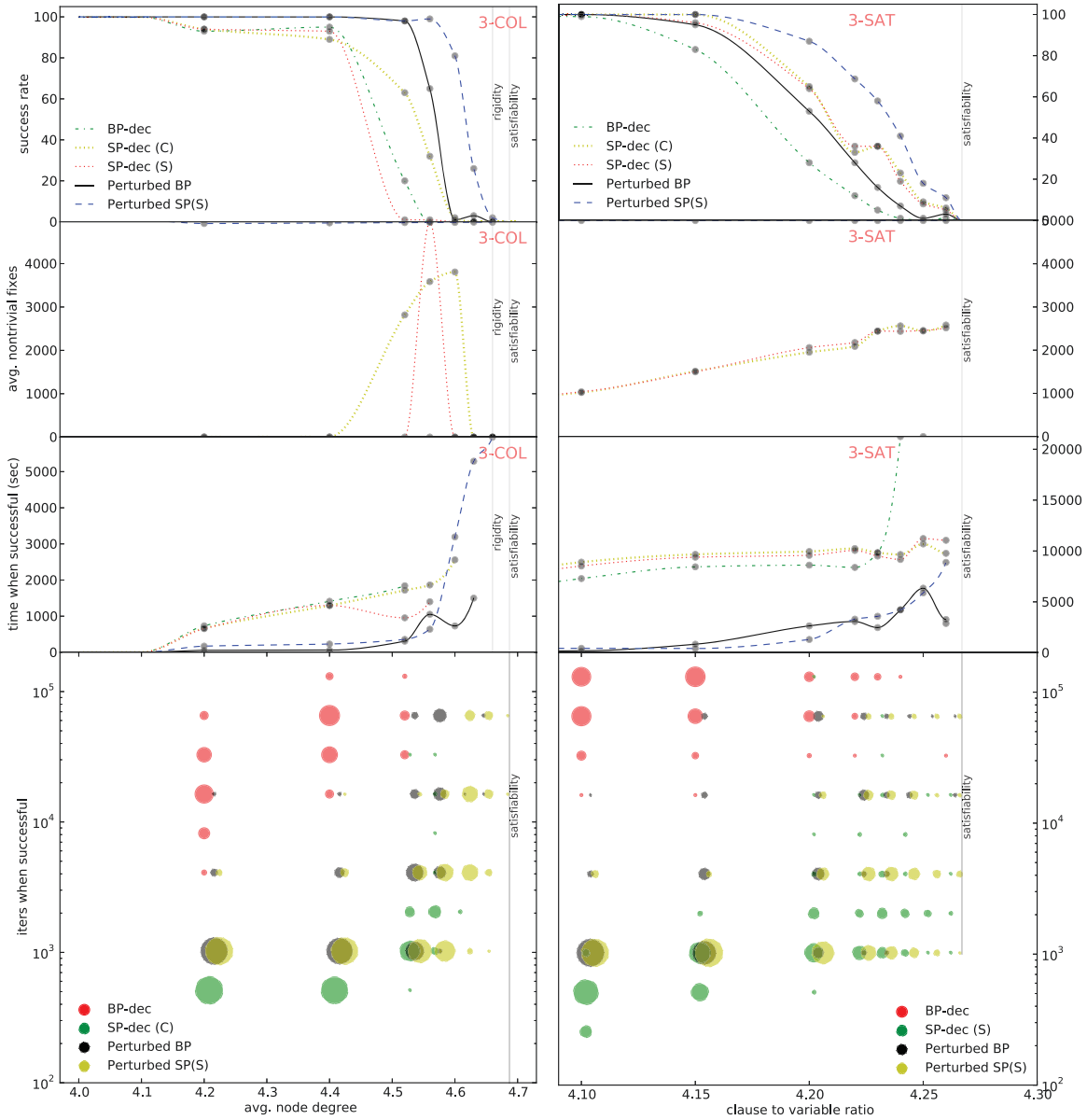


Figure 3.5: (first row) *Success-rate of different methods for 3-COL and 3-SAT for various control parameters.* ***(second row)*** *The average number of variables (out of $N = 5000$) that are fixed using SP-dec (C) and (S) before calling local search, averaged over 100 instances.* ***(third row)*** *The average amount of time (in seconds) used by the successful setting of each method to find a satisfying solution. For SP-dec(C) and (S) this includes the time used by local search.* ***(forth row)*** *The number of iterations used by different methods at different control parameters, when the method was successful at finding a solution. The number of iterations for each of 100 random instances is rounded to the closest power of 2. This does not include the iterations used by local search after SP-dec.*

this time includes the time used by local search. The final row of figure 3.5 shows the number of iterations used by each method at each level of difficulty over the successful instances. Note that this does not include the iterations of local search for SP-dec variations. Here the area of each disk is proportional to the frequency of satisfied instances with particular number of iterations for each control parameter and inference method[10].

Here we make the following observations:

(I) Perturbed BP is much more effective than BP-dec, while remaining ten to hundreds of time more efficient. (II) As the control parameter grows larger, the chance of requiring more iterations to satisfy the instance increases for all methods. (III) Although computationally very inefficient, BP-dec is able to find solutions for instances with larger control parameter than suggested by previous results (*e.g.*, [208]). (IV) For many instances where SP-dec(C) and (S) use few iterations, the variables are fixed to a trivial cluster $\widehat{\mathrm{p}}_i = (1, 1, \ldots, 1)$, in which all assignments are allowed. This is particularly pronounced for 3-COL. For instances in which non-trivial fixes are zero, the success rate is solely due to local search (*i.e.*, BP-dec). (V) While SP-dec(C) and SP-dec(S) have a similar performance for 3-SAT, SP-dec(C) significantly outperforms SP-dec(S) for 3-COL.

Table 3.1 reports the success-rate as well as the average of total iterations in the *successful* attempts of each method, where the number of iterations for SP-dec(C) and (S) is the sum of iterations used by the method plus the iterations of the following BP-dec. Here we observe that perturbed BP can solve most of easier instances using only $T = 1000$ iterations (*e.g.*, see perturb BP's result for 3-SAT at $\alpha = 4.$, 3-COL at $\alpha = 4.2$ and 9-COL at $\alpha = 33.4$). The results also show that most difficult instances (that require more time/iterations) for each method approximately correspond to the control parameter for which half of the instances are satisfied. Larger control parameters usually result in early failure in satisfiability.

Table 3.1 suggests that, as we speculated in section 3.2, SP-dec(C) is in general preferable to SP-dec(S), in particular when applied to the coloring problem. The most important advantage of Perturbed BP over SP-dec and Perturbed SP is that it can be applied to instances with large factor cardinality (*e.g.*, 10-SAT) and variable domains (*e.g.*, 9-COL). For example for 9-COL, the cardinality of each SP message is $2^9 = 512$, which makes SP-dec and Perturbed SP impractical. Here BP-dec is not even able to solve a single instance around the dynamical transition (as low as $\alpha = 33.4$) while perturbed BP satisfies all instances up to $\alpha = 34.1$.[11]

[10] The number of iterations are rounded to the closest power of two.

[11] Note that for 9-COL condensation transition happens after rigidity transition. So if we were able to find solutions after rigidity, it would have implied that condensation transition marks the onset of difficulty. However, this did not occur and similar to all other cases, Perturbed BP failed before rigidity transition.

Table 3.1: *Comparison of different methods on* $\{3,4\}$***-SAT*** *and* $\{3,4,9\}$***-COL****. For each method the success-rate and the average number of iterations (including local search) on successful attempts are reported. The approximate location of phase transitions are from [219, 330] .*

Problem	ctrl param α	BP-dec		SP-dec(C)		SP-dec(S)		Perturbed BP		Perturbed SP	
		avg. iters.	success rate	avg. iters.	success rate	avg. iters.	success rate	avg. iters.	success rate	avg. iters.	success rate
	3.86	dynamical and condensation transition									
	4.1	85405	99%	102800	100%	96475	100%	1301	100%	1211	100%
	4.15	104147	83%	118852	100%	111754	96%	5643	95%	1121	100%
	4.2	93904	28%	118288	65%	113910	64%	19227	53%	3415	87%
3-SAT	4.22	100609	12%	112910	33%	114303	36%	22430	28%	8413	69%
	4.23	123318	5%	109659	36%	107783	36%	18438	16%	9173	58%
	4.24	165710	1%	126794	23%	118284	19%	29715	7%	10147	41%
	4.25	N/A	0%	123703	9%	110584	8%	64001	1%	14501	18%
	4.26	37396	1%	83231	6%	106363	5%	32001	3%	22274	11%
	4.268	satisfiability transition									
	9.38	dynamical transition									
	9.547	condensation transition									
	9.73	134368	8%	119483	32%	120353	35%	25001	43%	11142	86%
4-SAT	9.75	168633	5%	115506	15%	96391	21%	36668	27%	9783	68%
	9.78	N/A	0%	83720	9%	139412	7%	34001	12%	11876	37%
	9.88	rigidity transition									
	9.931	satisfiability transition									
	4	dynamical and condensation transition									
	4.2	24148	93%	25066	94%	24634	94%	1511	100%	1151	100%
	4.4	51590	95%	52684	89%	54587	93%	1691	100%	1421	100%
	4.52	61109	20%	68189	63%	54736	1%	7705	98%	2134	98%
3-COL	4.56	N/A	0%	63980	32%	13317	1%	28047	65%	3607	99%
	4.6	N/A	0%	74550	2%	N/A	0%	16001	1%	18075	81%
	4.63	N/A	0%	N/A	0%	N/A	0%	48001	3%	29270	26%
	4.66	rigidity transition									
	4.66	N/A	0%	N/A	0%	N/A	0%	N/A	0%	40001	2%
	4.687	satisfiability transition									
	8.353	dynamical transition									
	8.4	64207	92%	72359	88%	71214	93%	1931	100%	1331	100%
4-COL	8.46	dynamical transition									
	8.55	77618	13%	60802	13%	62876	9%	3041	100%	5577	100%
	8.7	N/A	0%	N/A	0%	N/A	0%	50287	14%	N/A	0%
	8.83	rigidity transition									
	8.901	satisfiability transition									
	33.45	dynamical transition									
	33.4	N/A	0%	N/A	N/A	N/A	N/A	1061	100%	N/A	N/A
	33.9	N/A	0%	N/A	N/A	N/A	N/A	3701	100%	N/A	N/A
	34.1	N/A	0%	N/A	N/A	N/A	N/A	12243	100%	N/A	N/A
9-COL	34.5	N/A	0%	N/A	N/A	N/A	N/A	48001	6%	N/A	N/A
	35.0	N/A	0%	N/A	N/A	N/A	N/A	N/A	0%	N/A	N/A
	39.87	rigidity transition									
	43.08	condensation transition									
	43.37	satisfiability transition									

3.4 Clique-cover problem

The **K-clique-cover** $C = \{C_1, \dots, C_K\}$ for a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a partitioning of $\mathcal{V}$ to at most K cliques – *i.e.*, $\forall i,j,k \quad i,j \in C_k \Rightarrow (i,j) \in \mathcal{E}$.

$\mathbb{NP}$-completeness of K-clique-cover can be proved by reduction from K-coloring [159]: A K-clique-cover for $\mathcal{G}'$, the complement of $\mathcal{G}$ (*i.e.*, $\mathcal{G}' = (\mathcal{V}, \mathcal{E}' = \{(i,j) \mid (i,j) \notin \mathcal{E}\})$), is a K-coloring for $\mathcal{G}$, where all the nodes in the same clique of $\mathcal{G}'$ are allowed to have the same color in $\mathcal{G}$.

The relation between K-clique-cover and K-coloring extends to their factor-graphs. While in K-coloring, factors $\mathrm{f}_{\{i,j\}}(x_i, x_j) = \mathbb{1}(x_i \neq x_j) \quad \forall (i,j) \in \mathcal{E}$ ensure that the connected nodes have different colors, for k-clique-cover factors $\mathrm{f}_{\{i,j\}}(x_i, x_j) = \mathbb{1}(x_i \neq x_j) \quad \forall (i,j) \notin \mathcal{E}$ ensure that nodes that are not connected can not belong to the same clique. Here $x_i \in \{1, \dots, K\}$ represents the clique of node i.

The factors, $\mathrm{f}_{\{i,j\}}(x_i, x_j) = \mathbb{1}(x_i \neq x_j)$, in both K-clique-cover and K-coloirng are inverse Potts factors that allow efficient $O(K)$ calculation (see section 2.2.1). Using $\mathcal{E}(i,\cdot) = \{(i,j) \in \mathcal{E}\}$ to denote the set of edges adjacent to node i, the following claim states the complexity of BP updates.

Claim 3.4.1. *Each iteration of BP with variable-synchronous message update for K-clique-cover factor-graph is $O(K(N^2 - |\mathcal{E}|))$, while asynchronous message update is $O(K \sum_{i \in \mathcal{V}} (N - |\mathcal{E}(i,\cdot)|)^2)$.*

Proof. Here the complexity of calculating factor-to-variable message ($\widehat{\mathrm{p}}_{\{i,j\} \to i}$) is $O(K)$. Since there are $N^2 - \mathcal{E}$ factors (one for each edge in $\mathcal{G}'$) the total cost of factor-to-variable messages becomes $O(K(N^2 - |\mathcal{E}|))$.

The time complexity of each variable-to-factor message ($\widehat{\mathrm{p}}_{i \to \{i,j\}}$) is $O(K|\Delta i|)$, where Δi, Markov blanket of i in the factor-graph, is the set of nodes in $\mathcal{V}$ that are *not* adjacent to i in $\mathcal{G}$ – *i.e.*, $|\Delta i| = N - \mathcal{E}(i,\cdot)$. Using variable-synchronous update the total cost of variable-to-factor messages becomes $O(K \sum_{i \in \mathcal{V}} N - |\mathcal{E}(i,\cdot)|) = O(K(N^2 - 2|\mathcal{E}|))$. This means the cost of all messages in BP update is in the order of $O(K(N^2 - |\mathcal{E}|))$.

However, using asynchronous update, at each node i, we have to calculate $N - |\mathcal{E}(i,\cdot)|$ messages. Since each of them is $O(N - |\mathcal{E}(i,\cdot)|)$, the total cost of variable-to-factor dominates the cost of BP update which is $O(K \sum_{i \in \mathcal{V}} (N - |\mathcal{E}(i,\cdot)|)^2)$. □

Our experimental results for K-clique-cover are within the context of a binary-search scheme, as the sum-product reduction of the min-max formulation of K-clustering.

3.5 Dominating set and set cover

The K-**dominating set** of graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a subset of nodes $\mathcal{D} \subseteq \mathcal{V}$ of size $|\mathcal{D}| = K$ such that any node in $\mathcal{V} \setminus \mathcal{D}$ is adjacent to at least one member of $\mathcal{D}$ – *i.e.*, $\forall i \in \mathcal{V} \setminus \mathcal{D} \quad \exists j \in \mathcal{D} \; s.t. \; (i,j) \in \mathcal{E}$.

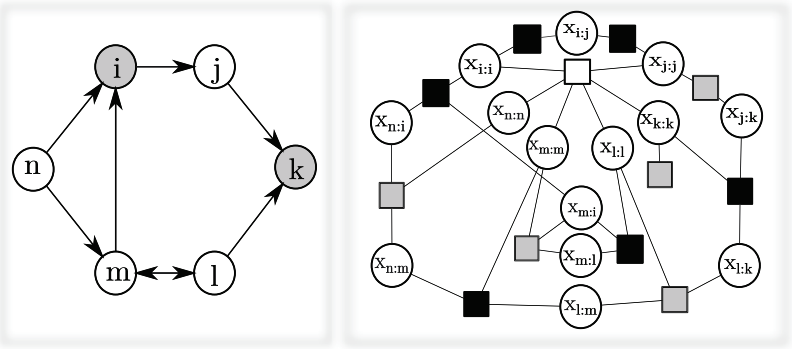


Figure 3.6: (left) *an induced 2-set-cover problem and the solution $\mathcal{D} = \{i,k\}$.* ***(right)*** *The factor-graph representation of the same problem, where leader factors are grey squares, consistency factors are in black and the K-of-N factor is in white.*

The dominating set problem is $\mathbb{NP}$-complete [107] and has simple reductions to and from set cover problem [158]. As we see, the factor-graph formulations of these problems are also closely related.

Given universe set $\mathcal{V}$ and a set of its subsets $\mathcal{S} = \{\mathcal{V}_1, \ldots, \mathcal{V}_M\}$ *s.t.* $\mathcal{V}_m \subseteq \mathcal{V}$, we say $\mathcal{C} \subseteq \mathcal{S}$ **covers** $\mathcal{V}$ iff each member of $\mathcal{V}$ is present in at least one member of $\mathcal{C}$ *–i.e.,* $\bigcup_{\mathcal{V}_m \in \mathcal{C}} \mathcal{V}_m = \mathcal{V}$. Now we consider a natural set-cover problem induced by any **directed** graph. Given a directed-graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, for each node $i \in \mathcal{V}$, define a subset $\mathcal{V}_i = \{j \in \mathcal{V} \mid (j,i) \in \mathcal{E}\}$ as the set of all nodes that are connected *to* i. Let $\mathcal{S} = \{\mathcal{V}_1, \ldots, \mathcal{V}_N\}$ denote all such subsets. An induced K-set-cover of $\mathcal{G}$ is a set $\mathcal{C} \subseteq \mathcal{S}$ of size K that covers $\mathcal{V}$. Equivalently, The **induced K-set-cover** of $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is a subset of vertices $\mathcal{D} \subseteq \mathcal{V}$, with $|\mathcal{D}| = K$, such that every node not in $\mathcal{D}$ is connected to at least one node in $\mathcal{D}$. For example in figure 3.6(left), $\mathcal{S} = \{\{n,m,i\},\{i,j\},\{j,k,l\},\{l,m,n\},\{n\}\}$ and its induced solution $\mathcal{C} = \{\{n,m,i\},\{j,k,l\}\}$ is indicated by grey nodes $\mathcal{D} = \{i,k\}$.

If we consider an undirected graph $\mathcal{G}$ as a directed graph with edges in both directions, then K-dominating set of $\mathcal{G}$ is equivalent to an induced K-set-cover problem on $\mathcal{G}$. Moreover given any K-set-cover problem instance $\mathcal{S} = \{\mathcal{V}_1, \ldots, \mathcal{V}_m\}$, we can construct a directed graph $\mathcal{G}$ such that the "induced" K-set-cover on $\mathcal{G}$ is equivalent to the given K-set-cover problem. For this, let $\mathcal{V} = (\bigcup_{\mathcal{V}_m \in \mathcal{S}} \mathcal{V}_m) \cup \{u_1, \ldots, u_M\}$ be the collection of nodes in $\mathcal{S}$ plus one node u_m per each subset $\mathcal{V}_m \in \mathcal{S}$. Now define the directed edges in $\mathcal{E}$ to connect every $i \in \mathcal{V}_m$ to its representative u_m. Moreover connect all representatives to each other in both directions – *i.e.,* $\mathcal{E} = \{(i,u_m) \mid \forall m, i \in \mathcal{V}_m\} \cup \{(u_m, u_{m'}) \mid \forall m \neq m'\}$. It is easy to show that the induced K-set-cover on this directed graph defines a set-cover for $\mathcal{S}$.

3.5.1 Factor-graph and complexity

For both problems we have one variable per edge $x_{i:j} \in \{0,1\}\ \forall (i,j) \in \mathcal{E}$. Note that the $\mathcal{G}$ for induced K-set-cover problem is a directed graph, while the $\mathcal{G}$ for the K-dominating-set is undirected. This is the only difference that affects the factor-graph representation of these two problems. Here,

$x_{i:j} = 1$ indicates that node $j \in \mathcal{D}$ and node i is associated with node j. Three types of constraint factors ensure the assignments to $x_{i:j}$ define a valid solution to K-dominating-set and induced K-set-cover:

• **Leader factors** ensure that each node i is associated with at least one node j (where $j = i$ is admissible). Let $\mathcal{E}^+(i,\cdot) = \{(i,j) \in \mathcal{E}\} \cup \{(i,i)\}$ be the set of edges leaving node i plus (i,i). Then

$$\mathsf{f}_{\mathcal{E}^+(i,\cdot)}(\underline{x}_{\mathcal{E}^+(i,\cdot)}) = \mathbb{1}((\sum_{(i,j)\in\mathcal{E}^+(i,\cdot)} x_{i:j}) \geq 1) \quad \forall i \in \mathcal{V} \tag{3.6}$$

is the leader factor associated with node i.

• **Consistency factors** ensure that if node j is selected as the leader by node i, node j also selects itself as leader:

$$\mathsf{f}_{\{i:j,j:j\}}(x_{i:j}, x_{j:j}) = \mathbb{1}(x_{i:j} = 0 \vee x_{j:j} = 0) \quad \forall (i,j) \in \mathcal{E} \tag{3.7}$$

An alternative form of this factor is a high-order factor that allows efficient $O(|\mathcal{E}(\cdot,i)|)$ factor-to-variable update

$$\mathsf{f}_{\mathcal{E}(\cdot,i)}(\underline{x}_{\mathcal{E}(\cdot,i)}) = \mathbb{1}(x_{i:i} = 1 \vee \sum_{(i,j)\in\mathcal{E}(\cdot,i)} x_{j:i} = 0) \quad \forall i \in \mathcal{V} \tag{3.8}$$

• **At most K-of-N factor** ensures that at most K nodes are selected as leaders ($|\mathcal{D}| \leq K$):

$$\mathsf{f}_{\{i:i,j:j,\ldots,l:l\}}(\underline{x}_{\{i:i,j:j,\ldots,l:l\}}) = \mathbb{1}(\sum_{i\in\mathcal{V}} x_{i:i} \leq K) \tag{3.9}$$

Figure 3.6 shows an example of induced K-set-cover problem and its corresponding factor-graph. In section 2.2.1 we saw that it is possible to calculate sum-product factor-to-variable BP messages for leader factors in $O(|\mathcal{E}(i,\cdot)|)$ and each at-most-K-of-N factor in $O(KN)$. This cost for consistency factors is $O(1)$ for the pairwise and $O(|\mathcal{E}(\cdot,i)|)$ for the alternative formulation.

Claim 3.5.1. *The time-complexity of message passing for the factor-graph above depending on the update schedule is $O(|\mathcal{E}|+KN)$ for factor-synchronous (f-sync; see section 2.1.1) update and $O(KN^2 + \sum_{i\in\mathcal{V}} |\mathcal{E}(i,\cdot)|^2 + |\mathcal{E}(\cdot,i)|^2)$ for asynchronous update.*

Proof. We assume the consistency factors are in the higher order form of equation (3.8). Here, each variable $x_{i:j}$ is connected to at most three factors and therefore the cost of variable-to-factor messages is $O(1)$. If we calculate factor-to-variable messages simultaneously, the cost is $O(\sum_{i\in\mathcal{V}} |\mathcal{E}(i,\cdot)|)$ for leader and $O(\sum_{i\in\mathcal{V}} |\mathcal{E}(\cdot,i)|)$, giving a total of $O(|\mathcal{E}|)$. Adding this to the cost of K-of-N factor the total cost per iteration of BP is $O(|\mathcal{E}| + KN)$.

On the other hand, if we update each factor-to-variable separately, the previous costs are multiplied by $|\partial \mathrm{I}|$, which gives $O(KN^2 + \sum_{i\in\mathcal{V}} |\mathcal{E}(i,\cdot)|^2 + |\mathcal{E}(\cdot,i)|^2)$. □

3.6 Clique problem, independent set and sphere packing

Given graph $\mathcal{G}$, the **K-clique problem** asks whether $\mathcal{G}$ contains a clique of size at least K. The K-clique problem is closely related to **K-independent-set**. Given graph $\mathcal{G} = (\mathcal{V},\mathcal{E})$, the K-independent set problem asks whether $\mathcal{V}$ contains a subset of size at least K, s.t. there is no connection between nodes in $\mathcal{D}$ – *i.e.*, $\forall i,j \in \mathcal{D} \quad (i,j) \notin \mathcal{E}$. The relation between K-clique problem and K-independent-set is analogous to the connection between K-coloring and K-clique-cover problems: the K-clique problem on $\mathcal{G}$ is equivalent to K-independent-set problem on its complement $\mathcal{G}'$.

K-independent-set is in turn equivalent to (N-K)-**vertex cover** problem. A vertex cover $\mathcal{D}' \subseteq \mathcal{V}$ is a subset of nodes such that each edge is adjacent to at least one vertex in $\mathcal{D}'$. It is easy to see that $\mathcal{D}$ is an independent set iff $\mathcal{D}' = \mathcal{V} \setminus \mathcal{D}$ is a vertex cover. Therefore our solution here for independent set directly extends to vertex cover.

K-packing is a special case of sub-graph isomorphism, which asks whether a graph $\mathcal{G}_1$ is a sub-graph of $\mathcal{G}_2$. For packing, $\mathcal{G}_2$ is the main graph ($\mathcal{G}$) and $\mathcal{G}_1$ is the complete graph of size K (see section 5.3) K-independent-set and K-clique problems are also closely related to **sphere packing** and finding nonlinear codes. To better motivate these problems, here we start with the problem of K-packing formulated as min-max problem in (several) factor-graphs. We then show that the sum-product reduction of K-packing is K-independent-set and K-clique problems. By doing this we simultaneously introduce message passing solutions to all three problems.

Given a symmetric distance matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$ between N data-points and a number of **code-words** K, the **K-packing** problem is to choose a subset of K points such that the minimum distance $\mathbf{A}_{i,j}$ between any two code-words is maximized. Here we introduce two different factor-graphs such that min-max inference obtains the K-packing solution.

In order to establish the relation between the min-max problem and the CSPs above, we need the notion of y-neighbourhood graph

Definition 3.6.1. The **y-neighborhood graph** for (distance) matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$ is defined as the graph $\mathcal{G}(\mathbf{A},y) = (\mathcal{V},\mathcal{E})$, where $\mathcal{V} = \{1,\dots,N\}$ and $\mathcal{E} = \{(i,j) \mid \mathbf{A}_{i,j} \leq y\}$.

3.6.1 Binary variable factor-graph

Let binary variables $\underline{x} = \{x_1,\dots,x_N\} \in \{0,1\}^N$ indicate a subset of variables of size K that are selected as code-words. We define the factors such that the min-max assignment

$$\underline{x}^* = \arg_{\underline{x}} \min \max_{\mathrm{I}\in\mathcal{F}} \mathrm{f}_{\mathrm{I}}(\underline{x}_{\mathrm{I}})$$

is the K-packing solution. Define the factor-graph with the following two types of factors:
• **K-of-N factor** (section 2.2.1)

$$\mathsf{f}_{\mathcal{N}}(\underline{x}) \;=\; 1((\sum_{i\in\mathcal{N}} x_i) = K)$$

ensures that K code-words are selected. Recall that definition of 1(.) depends on the semiring (equation (1.20)). The K-packing problem is defined on the min-max semiring. However, since we plan to solve the min-max inference by sum-product reductions, it is important to note that the p_y reduction of 1(.) for any y is 1(.) as defined for sum-product ring.
• **Pairwise factors** are only effective if both x_i and x_j are non-zero

$$\mathsf{f}_{\{i,j\}}(x_i,x_j) = \min\Big(1(x_i = 0 \vee x_j = 0),\ \max\Big(1(x_i = 1 \wedge x_j = 1),\ -\mathbf{A}_{i,j}\Big)\Big) \quad \forall i,j \tag{3.10}$$

where the tabular form is simply

x_i \ x_j	0	1
0	$-\infty$	$-\infty$
1	$-\infty$	$-\mathbf{A}_{i,j}$

Here the use of $-\mathbf{A}_{i,j}$ is to convert the initial max-min objective to min-max.[12] Recall that $\mathcal{G}(\mathbf{A},y)$ defines a graph based on the distance matrix $\mathbf{A}$, s.t. two nodes are connected iff their distance is "not larger than" y. This means $\mathcal{G}(-\mathbf{A},-y)$ corresponds to a graph in which the connected nodes have a distance of "at least" y.

Proposition 3.6.1. *The p_y-reduction of the K-packing factor-graph above for the distance matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$ defines a uniform distribution over the cliques of $\mathcal{G}(-\mathbf{A},-y)$ of size K.*

Proof. Since $\mathsf{p}_y(\underline{x})$ is uniform over its domain it is enough to show that:
• *Every clique of size K in $\mathcal{G}(-\mathbf{A},-y)$ corresponds to a unique assignment $\underline{x}^*$ with $\mathsf{p}_y(\underline{x}^*) > 0$:*
Given a clique $C \subseteq \mathcal{V}$ of size K in $\mathcal{G}(-\mathbf{A},-y) = (\mathcal{V},\{(i,j) \mid \mathbf{A}_{i,j} \geq y\})$, define $\underline{x}^* = \{x_i = ident(i \in C) \mid i \in \mathcal{V}\}$. It is easy to show that $\mathsf{p}_y(\underline{x}^*) > 0$. For this we need to show that all the constraint factors in p_y are satisfied. The K-of-N factor is trivially satisfied as $|C| = K$. The p_y-reduction of the pairwise factor of equation (3.10) becomes

$$\mathsf{f}_{\{i,j\}}(x_i,x_j) = 1(x_i = 0 \vee x_j = 0) + \Big(1(x_i = 1 \wedge x_j = 1)1(-\mathbf{A}_{i,j} \leq -y)\Big) \quad \forall i,j \tag{3.11}$$

where we replaced min and max with + and . operators of the sum-product semiring and thresholded $-\mathbf{A}_{i,j}$ by $-y$. To see that all pairwise constraint factors are satisfied consider two cases: (I)

[12] The original objective is max-min because it aims to maximize the minimum distance between any two code-words.

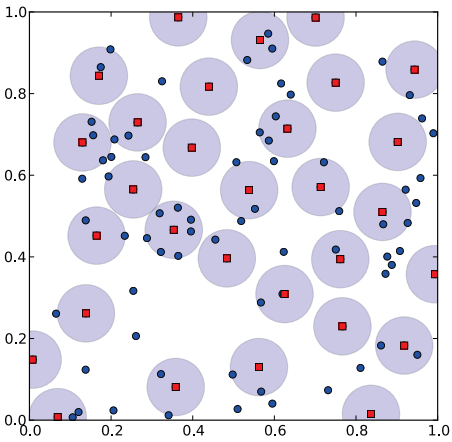


Figure 3.7: *Using message passing to choose $K = 30$ out of $N = 100$ random points in the Euclidean plane to maximize the minimum pairwise distance (with $T = 500$ iterations for PBP). Touching circles show the minimum distance*

nodes $i, j \in C$, and therefore $x_i^* = x_j^* = 1$. This also means i and j are connected and definition of $\mathcal{G}(-\mathbf{A}, -y)$, this implies $\mathbf{A}_{i,j} \geq y$. Therefore the second term in factor above evaluates to one. (II) either i or j are not in C, therefore the first term evaluates to one. Since both pairwise factors and the K-of-N factors for $\underline{x}^*$ are non-zero $\mathrm{p}_y(\underline{x}^*) > 0$.

• *every assignment $\underline{x}^*$ with $\mathrm{p}_y(\underline{x}^*) > 0$ corresponds to a unique clique of size K in $\mathcal{G}(-\mathbf{A}, -y)$:*
equation (3.11) implies $x_i = 1 \wedge x_j = 1 \quad \Rightarrow \quad \mathbf{A}_{i,j} \geq y$. On the other hand, $\mathrm{p}_y(\underline{x}) > 0$ means K-of-N factor is satisfied and therefore exactly K variables x_i are nonzero. Therefore the index of these variables identifies subset of nodes in $\mathcal{G}(-\mathbf{A}, -y)$ that are connected (because $\mathbf{A}_{i,j} \geq y$), forming a clique. □

Claim 3.6.2. *The time-complexity of each iteration of sum-product BP for the factor-graph above is*
- *$O(NK + |\mathcal{E}|)$ for variable and factor synchronous update.*
- *$O(N^2K)$ for factor-synchronous (f-sync) message update.*
- *$O(N^3)$ for completely asynchronous update.*

Proof. The cost of f-sync calculation of factor-to-variable messages for the K-of-N factor is $O(NK)$ and the cost of factor-to-variable messages for pairwise factors is $O(1)$. Since there are $|\mathcal{E}|$ such factor, this cost evaluates to $O(NK + |\mathcal{E}|)$. Since each node is adjacent to $|\mathcal{E}(i, \cdot)|$ other nodes, the variable-synchronous update of variable-to-factor messages is $O(\sum_{i \in \mathcal{V}} |\mathcal{E}(i, \cdot)|) = O(|\mathcal{E}|)$, which gives a total time-complexity of $O(NK + |\mathcal{E}|)$.

In asynchronous update, the cost of factor-to-variable messages for the K-of-N factor is $O(N^2K)$ as we need to calculate each message separately. Moreover, updating each variable-to-factor message is $O(|\mathcal{E}(i, \cdot)|)$, resulting a total of $O(\sum_{i \in \mathcal{V}} |\mathcal{E}(i, \cdot)|^2) = O(N^3)$. Since $K \leq N$, when all the updates are asynchronous, this cost subsumes the factor-to-variable cost of $O(N^2K)$. □

A corollary is that the complexity of finding the approximate min-max solution by sum-product reduction is $O((NK + |\mathcal{E}|)\log(N))$ using synchronous update. Figure 3.7 shows an example of the solution found by message passing for K-packing with Euclidean distance.

3.6.2 Categorical variable factor-graph

Define the K-packing factor-graph as follows:Let $\underline{x} = \{x_1, \ldots, x_K\}$ be the set of K variables where $x_i \in \{1, \ldots, N\}$. For every two distinct points $1 \leq i < j \leq K$ define the factor

$$\mathrm{f}_{\{i,j\}}(x_i, x_j) = \max\Big(-\mathbf{A}_{x_i,x_j}, 1(x_i \neq x_j)\Big) \tag{3.12}$$

Here each variable represents a code-word and this factor ensures that code-words are distinct. Moreover if x_i and x_j are distinct, $\mathrm{f}_{\{i,j\}}(x_i, x_j) = -\mathbf{A}_{x_i,x_j}$ is the distance between the nodes that x_i and x_j represent.

The tabular form of this factor is

		x_j				
		1	2	$\cdots$	$N-1$	N
x_i	1	∞	$-\mathbf{A}_{1,2}$	$\cdots$	$-\mathbf{A}_{1,N-1}$	$-\mathbf{A}_{1,N}$
	2	$-\mathbf{A}_{2,1}$	∞	$\cdots$	$-\mathbf{A}_{2,N-1}$	$-\mathbf{A}_{2,N}$
	$\cdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$
	$N-1$	$-\mathbf{A}_{N-1,1}$	$-\mathbf{A}_{N-1,2}$	$\cdots$	∞	$-\mathbf{A}_{N-1,N}$
	N	$-\mathbf{A}_{N,1}$	$-\mathbf{A}_{N,2}$	$\cdots$	$-\mathbf{A}_{N,N-1}$	∞

The following proposition relates this factor-graph to K-clique problem.

Proposition 3.6.3. *The p_y-reduction of the K-packing factor-graph above for the distance matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$ defines a uniform distribution over the cliques of $\mathcal{G}(-\mathbf{A}, -y)$ of size K.*

Proof. Since p_y defines a uniform distribution over its support, it is enough to show that any clique of size K over $\mathcal{G}(-\mathbf{A}, -y)$ defines a unique set of assignments all of which have nonzero probability ($\mathrm{p}_y(\underline{x}) > 0$) and any assignment $\underline{x}$ with $\mathrm{p}_y(\underline{x}) > 0$ defines a unique clique of size at least K on $\mathcal{G}(-\mathbf{A}, -y)$. First note that the basic difference between $\mathcal{G}(\mathbf{A}, y)$ and $\mathcal{G}(-\mathbf{A}, -y)$ is that in the former all nodes that are connected have a distance of at most y while in the later, all nodes that have a distance of at least y are connected to each other. Consider the p_y-reduction of the pairwise factor of equation (3.12)

$$\begin{aligned}\mathrm{f}_{\{i,j\}}(x_i, x_j) &= 1\Big(\max(-\mathbf{A}_{x_i,x_j}, 1(x_i \neq x_j)) \leq y\Big) \\ &= 1(-\mathbf{A}_{x_i,x_j} \leq y \wedge x_i \neq x_j)\end{aligned} \tag{3.13}$$

where, basically, we have replaced max from min-max semiring with $\otimes$ operator of the sum-product semiring and thresholded $\mathbf{A}_{x_i,x_j}$.

• *Any clique of size K in $\mathcal{G}(-\mathbf{A},-y)$, defines K unique assignments, such that for any such assignment $\underline{x}^*$, $\mathrm{p}_y(\underline{x}^*) > 0$:*
For a clique $C = \{c_1,\ldots,c_K\} \subseteq \mathcal{V}$ of size K, define $x_i^* = c_{\pi(i)}$, where $\pi : \{1,\ldots,K\} \to \{1,\ldots,K\}$ is a permutation of nodes in clique C. Since there are K such permutations we may define as many assignments $\underline{x}^*$. Now consider one such assignment. For every two nodes x_i^* and x_j^*, since they belong to the clique C in $\mathcal{G}(-\mathbf{A},-y)$, they are connected and $\mathbf{A}_{x_i,x_j} \geq y$. This means that all the pairwise factors defined by equation (3.13) have non-zero values and therefore $\mathrm{p}_y(\underline{x}) > 0$.

• *Any assignment $\underline{x}^*$ with $\mathrm{p}_y(\underline{x}^*) > 0$ corresponds to a unique clique of size K in $\mathcal{G}(-\mathbf{A},-y)$:* Let $C = \{x_1^*,\ldots,x_K^*\}$. Since $\mathrm{p}_y(\underline{x}^*) > 0$, all pairwise factors defined by equation (3.13) are non-zero. Therefore $\forall i,j \neq i \quad \mathbf{A}_{x_i,x_j} \geq y$, which means all x_i and x_j are connected in $\mathcal{G}(-\mathbf{A},-y)$, forming a clique of size K. □

To put simply, in acquiring the p_y-reduction, we set the values in the table form above to zero if their value is less than y and set them to one otherwise. The resulting factor-graph, defines a distribution $\mathrm{p}_y(\underline{x})$, s.t. $\mathrm{p}_y(\underline{x}) > 0$ means $\underline{x}$ defines a clique of size K in a graph $\mathcal{G}(-\mathbf{A},-y)$ which connects nodes with distance larger than y.

Claim 3.6.4. *Each iteration of BP for this factor-graph with pairwise factors is $O(N^2K^2)$, for synchronized update and $O(N^2K^3)$ for asynchronous update. Using the sum-product reduction of min-max inference this suggests a $O(N^2K^2\log(N))$ (for sync. update) and $O(N^2K^3\log(N))$ (for asynchronous update) procedure for K-packing problem.*

Proof. (claim 3.6.4 on page 95) Since the factors are not sparse, the complexity of calculating a single factor-to-variable message is $O(|\mathcal{X}_\mathrm{I}|) = O(N^2)$, resulting in $O(N^2K^2)$ cost per iteration of variable-synchronous update for BP. However if we update each message separately, since each message update costs $O(N^2K)$, the total cost of BP per iteration is $O(N^2K^3)$

Since the diversity of pairwise distances is that of elements in $\mathbf{A}$ – *i.e.*, $|\mathcal{Y}| = O(N^2)$ – the general cost of finding an approximate min-max solution by message passing is $O(N^2K^2\log(N))$ for sync. message update and $O(N^2K^3\log(N))$ for async. update. □

The p_y-reduction of our second formulation was first proposed by [252] to find non-linear binary codes. The authors consider the Hamming distance between all binary vectors of length n (*i.e.*, $N = 2^n$) to obtain binary codes with known minimum distance y. As we saw, this method is $O(N^2K^2) = O(2^{2n}K^2)$ –*i.e.*, grows exponentially in the number of bits n. In the following section, we introduce a factor-graph formulation specific to categorical variables with Hamming distance whose message passing complexity is polynomial in $n = \log(N)$. Using this formulation we are able find optimal binary and ternary codes where both n and y are large.

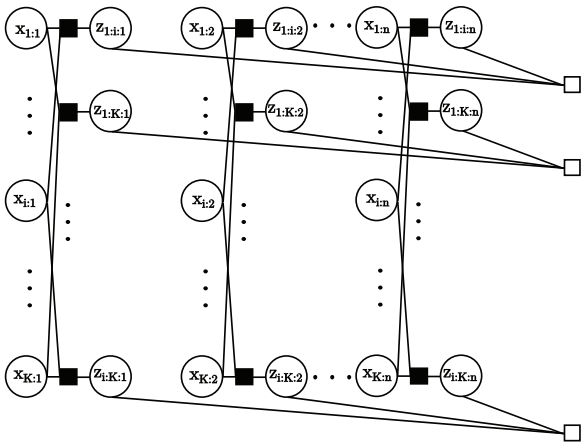


Figure 3.8: *The factor-graph for sphere-packing with Hamming distances, where the distance factors are white squares and z-factors are in black.*

3.6.3 Efficient Sphere packing with Hamming distance

Our factor-graph defines a distribution over the K binary vectors of length n such that the distance between every pair of binary vectors is at least y. [13]

To better relate this to K-clique problem, consider 2^n binary code-words of length n as nodes of a graph and connect two nodes iff their Hamming distance is at least y. Finding a K-clique in this graph is equivalent to discovery of so-called **nonlinear binary codes** – a fundamental problem in information theory (*e.g.*, see [75, 193]). Here assuming y is an odd number, if at most $\frac{y-1}{2}$ digits of a code-word are corrupted (*e.g.*, in communication), since every pair of codes are at least y digits apart, we can still recover the uncorrupted code-word. The following is a collection of $K = 12$ ternary code-words of length $n = 16$, obtained using the factor-graph that we discuss in this section, where every pair of code-words are different in at least $y = 11$ digits.

2	1	2	1	0	1	1	2	2	1	2	1	2	1	0	2
1	1	1	2	1	0	0	2	2	1	1	1	0	2	1	0
0	0	1	2	0	1	2	0	2	1	2	2	1	0	2	0
0	0	0	2	1	0	1	2	1	0	1	0	2	1	0	1
2	2	0	2	2	2	1	1	2	0	2	2	1	2	1	2
0	2	0	0	0	2	0	2	0	0	2	1	0	0	0	0
1	1	2	0	0	1	2	2	0	0	1	0	1	2	2	1
2	0	2	0	2	1	0	1	1	2	0	1	1	1	2	0
0	2	1	1	1	1	1	1	0	1	0	0	0	0	1	2
1	0	0	1	0	2	0	0	1	2	0	0	2	2	1	2
1	2	2	1	1	0	2	1	1	2	2	2	2	0	2	1
1	0	1	0	2	2	2	1	0	1	1	2	2	1	0	2

[13] For convenience we restrict this construction to the case of binary vectors. A similar procedure may be used to find maximally distanced ternary and q-ary vectors, for arbitrary q.

The construction of this factor-graph is more involved than our previous constructions. The basic idea to avoid the exponential blow up is to have one variable per *digit* of each code-word (rather than one variable per code-word). Then for each pair of code-words we define an auxiliary binary vector of the same length, that indicates if the two code-words are different in each digit. Finally we define an at-least-y-of-n constraint over each set of auxiliary vectors that ensures every two pair of code-words are at least different in y digits. Figure 3.8 shows this factor-graph.

More specifically, let $\underline{x} = \{\underline{x}_{1:}, \ldots, \underline{x}_{K:}\}$ be a set of binary vectors, where $\underline{x}_{i:} = \{x_{i:1}, \ldots, x_{i:n}\}$ represents the i^{th} binary vector or code-word. Additionally for each two code-words $1 \leq i < j \leq K$, define an auxiliary binary vector $\underline{z}_{i:j:} = \{z_{i:j:1}, \ldots, z_{i:j:n}\}$ of length n.

For each distinct pair of binary vectors $\underline{x}_{i:}$ and $\underline{x}_{j:}$, and a particular digit $1 \leq k \leq n$, the auxiliary variable is constrained to be $z_{i:j:k} = 1$ iff $x_{i:k} \neq x_{j:k}$. Then we define an at-least-y-of-n factor over $\underline{z}_{i:j:}$ for every pair of code-words, to ensure that they differ in at least y digits.

The factors are defined as follows

• **Z-factors:** For every $1 \leq i < j \leq K$ and $1 \leq k \leq n$, define

$$\mathrm{f}_{\{i:k,j:k,i:j:k\}}(x_{i:k}, x_{j:k}, z_{i:j:k}) = \mathbb{1}\Big(\big((x_{i:k} \neq x_{j:k}) \wedge z_{i:j:k} = 1\big) \vee \big((x_{i:k} = x_{j:k}) \wedge z_{i:j:k} = 0\big)\Big)$$

This factor depends on three binary variables, therefore we can explicitly define its tabular form containing $2^3 = 8$ possible inputs. Here the only difference between binary and ternary (and q-ary) codes in general is in the tabular form of this factor. For example for ternary codes, the tabular for of this factor is a $3 \times 3 \times 2$ array ($z_{i:j:k}$ is always binary).

• **Distance-factors:** For each $\underline{z}_{i:j:}$ define at-least-y-of-n factor (section 2.2.1):

$$\mathrm{f}_{i:j:}(\underline{z}_{i:j:}) = \mathbb{1}\Big(\sum_{1 \leq k \leq n} z_{i:j:k} \geq y\Big)$$

Claim 3.6.5. *Each iteration of sum-product BP over this factor-graph is*

– $O(K^2 ny)$ *for variable and factor synchronous update.*

– $O(K^2 n^2 y)$ *for variable-sync update.*

– $O(K^3 n + K^2 n^2 y)$ *for completely asynchronous BP update.*

Proof. (claim 3.6.5 on page 97) We first consider the complexity of variable-to-factor updates: Each auxiliary variable $z_{i:j:k}$ is connected to three factors and each $x_{i:j}$ is connected to $O(K)$ z-factors. Since there are $O(nK)$, $x_{i:j}$ variables a variable-sync. update of variable-to-factor messages is $O(nK^2)$, while async. update is $O(nK^3)$.

Next we consider two possibilities of factor-to-variable updates: We have $O(nK^2)$ z-factors, and factor-to-variable update for each of them is $O(1)$, this cost is subsumed by the minimum variable-to-factor cost. The factor-graph also has $O(K^2)$ distance factors, where the cost of each

factor-to-variable update is $O(ny)$. Since $|\partial \mathrm{I}| = K$ for a distance factor, a factor-sync. update is $O(K^2 ny)$ in total, while an async. update is $O(K^2 n^2 y)$.

Adding the cost of variable-to-factor and factor-to-variable in different scenarios we get the time-complexities stated in the claim. □

The following table reports some optimal binary codes (including codes with large number of bits n) from [193], recovered using this factor-graph. We used Perturbed BP with variable-sync update and $T = 1000$ iterations to find an assignment $\underline{x}^*, \underline{z}^*$ with $\mathrm{p}(\underline{x}^*, \underline{z}^*) > 0$.

n	K	y	n	K	y	n	K	y	n	K	y
8	4	5	11	4	7	14	4	9	16	6	9
17	4	11	19	6	11	20	8	11	20	4	13
23	6	13	24	8	13	23	4	15	26	6	15
27	8	15	28	10	15	28	5	16	26	4	17
29	6	17	29	4	19	33	6	19	34	8	19
36	12	19	32	4	21	36	6	21	38	8	21
39	10	21	35	4	23	39	6	23	41	8	23
39	4	25	43	6	23	46	10	25	47	12	25
41	4	27	46	6	27	48	8	27	50	10	27
44	4	29	49	6	29	52	8	29	53	10	29

3.7 Optimization variations of CSPs

This section briefly reviews the optimization variations of the CSPs, we have studied so far. The optimization version of satisfiability is known as **max-SAT** or maximum weighted SAT, where each clause has a weight, and the objective is to maximize the weighted sum of satisfied clauses. Here, simply using factors $\mathrm{f}_\mathrm{I}(\underline{x}_\mathrm{I}) : \mathcal{X}_\mathrm{I} \to \{0, -w_\mathrm{I}\}$, where w_I is the positive weight of clause I, min-sum BP will attempt to find the max-SAT solution. Note that here f_I is not a "constraint" factor anymore (see definition 1.3.2). Alternative approaches using variations of energetic survey propagation has also been used to improve max-SAT results [63, 64]. A less studied optimization variation of satisfiability is adversarial SAT which corresponds to min-max-sum inference [54].

For minimum coloring – a.k.a. **chromatic number** – and **minimum clique-cover** problem, since the optimal value $1 \leq K^* \leq K_{\max}$ is bounded, by access to an oracle for the decision version (or an incomplete solver [160] such as message passing), we can use **binary search** to find the minimum K in $O(\tau \log(K_{\max}))$ time, where the decision problem has a $O(\tau)$ time-complexity. In particular, since the chromatic number is bounded by the maximum degree [51], approximating the chromatic number using binary search and message passing gives a $O(\log(\max_i(|\mathcal{E}(i,\cdot)|))\ \max_i(|\mathcal{E}(i,\cdot)|)\ |\mathcal{E}|)$ time procedure.

The same binary-search approach can be used for minimum dominating-set, minimum set-cover, maximum clique, maximum independent set and minimum vertex cover. However, these optimization variations also allow a more efficient and direct approach. Note that both min-max variation and the minimization (or maximization) variations of these problems use binary search. For example both minimum clique-cover and K-clustering (see section 4.3) can be solved using binary-search over K-clique-cover decision problem. The parameter of interest for binary search is K in the former case, and y in the later case, where y is a threshold distance that defines connectivity in $\mathcal{G}$ (see definition 3.6.1). However, often both variations (*i.e.*, min-max and minimization or maximization over K) also allow direct message passing solutions.

For **minimum dominating-set and set-cover**, we replace the sum-product semiring with min-sum semiring and drop the K-of-N factor. Instead, a local factor $\mathsf{f}_{i:i}(x_{i:i}) = -x_{i:i}w_i$ gives a weight to each node $i \in \mathcal{V}$. Here, the min-sum inference seeks a subset of nodes that form a dominating set and have the largest sum of weights $\sum_{i\in\mathcal{D}} w_i$. Also note that by changing the semiring to min-sum, the identity functions $1(.)$ change accordingly so that the leader factor and consistency constraints remain valid. This gives an efficient $O(|\mathcal{E}|)$ synchronous procedure for minimum set-cover and minimum dominating-set. This problem and the resulting message passing solution are indeed a variation of K-medians and affinity propagation respectively (see section 4.1).

The same idea applies to **maximum clique and maximum independent set**: as an alternative to fixing or maximizing the "size" of a clique or an independent set, we may associate each node with a weight $w_i \ \forall i \in \mathcal{V}$ and seek a subset of nodes that form an independent set (or a clique) with maximum weight. Sanghavi et al. [274] study the max-product message passing solution to this problem and its relation to its LP-relaxation. In particular they show that starting from uniform messages, if BP converges, it finds the solution to LP relaxation.

Here we review their factor-graph for maximum independent set using min-sum inference. Let $\underline{x} = \{x_1, \ldots, x_N\} \in \{0,1\}^N$ be a set of binary variables, one for each node in $\mathcal{V}$, where $x_i = 1$ means $i \in \mathcal{D}$, the independent set.

• **Local factors** capture the cost (negative weight) for each node and is equal to $-w_i$ if $x_i = 1$ and zero otherwise

$$\mathsf{f}_i(x_i) \quad = \quad \min(-w_i, 1(x_i = 0)) \quad \forall i \in \mathcal{V}$$

• **Pairwise factors** ensure that if $(i,j) \in \mathcal{E}$, then either $x_i = 0$ or $x_j = 0$

$$\mathsf{f}_{\{i,j\}}(\underline{x}_{\{i,j\}}) \quad = \quad 1(x_i = 0 \vee x_j = 0) \quad \forall (i,j) \in \mathcal{E}$$

It is easy to see that using a variable synchronous update, message passing can be performed very efficiently in $O(|\mathcal{E}|)$.

Weigt and Zhou [309] (also see [333]) propose an interesting approach to **minimum vertex cover** using energetic survey propagation. Here, again $\underline{x} = \{x_1, \ldots, x_N\} \in \{0,1\}^N$ has one binary variable per node $i \in \mathcal{V}$ and a pairwise factor ensure that all edges are covered by at least one node in the cover

$$\mathsf{f}_{\{i,j\}}(\underline{x}_{\{i,j\}}) \quad = \quad 1(x_i = 1 \vee x_j = 1) \quad \forall (i,j) \in \mathcal{E}$$

Using the xor-and semiring, the resulting fixed points of warning propagation reveal minimal (but not necessarily optimal) vertex covers. The authors then suggest using survey propagation, and decimation to find the warning propagation fixed points with lowest energy (*i.e.*, smallest size of cover).

Chapter 4

Clustering

Clustering of a set of data-points is a central problem in machine learning and data-mining [4, 102, 180, 184, 191, 230, 236, 248, 251, 266, 318, 322]. However many interesting clustering objectives, including the problems that we consider in this section are $\mathbb{NP}$-hard.

In this section we present message passing solutions to several well-known clustering objectives including K-medians, K-clustering, K-centers and modularity optimization. Message passing has also been used within Expectation Maximization to obtain some of the best results in learning stochastic block models (a hidden variable model for clustering) [82]. The message passing solution to K-medians and its generalization, clustering by shallow trees are proposed by other researchers. However, for completeness, we review their factor-graphs in sections 4.1 and 4.2. expressing K-clustering and K-centers problems as min-max inference problems on factor-graphs in sections 4.3 and 4.4.

4.1 K-medians

Given a symmetric matrix of pairwise distances $\mathbf{A} \in \mathbb{R}^{N \times N}$ between N data-points, and a number of clusters K, K-medians seeks a partitioning of data-points into K clusters, each associated with a cluster center, s.t. the sum of distances from each data-point to its cluster center is minimized.

This problem is $\mathbb{NP}$-hard; however, there exists several approximation algorithms for metric distances [15, 58]. Here we present the binary variable factor-graph [112] for a slightly different version of this objective, proposed by Frey and Dueck [102]. The simplified form of min-sum BP messages in this factor-graph is known as **affinity propagation**. Here, instead of fixing the number of clusters, K, the objective is modified to incorporate each cluster's cost to become a center. This cost is added to the sum of distances between the data-points and their cluster centers and used to decide the number of clusters at run-time.

Let $\mathbf{A}$ be the distance matrix of a directed graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $(i,j) \notin \mathcal{E} \Leftrightarrow \mathbf{A}_{i,j} = \infty$.

Moreover let $\mathbf{A}_{i,i}$ denote the willingness of node i to be the center of a cluster. A simple heuristic is to set this value uniformly to $\text{median}_{i\neq j}\mathbf{A}_{i,j}$.

Define $\underline{x} = \{x_{i:j}, \forall\ (i,j) \in \mathcal{E}\}$ as a set of binary variables – one per each directed edge (i,j) – where $x_{i:j} \in \{0,1\}$ indicates whether node i follows node j as its center. Here node i can follow itself as center. The following factors define the cost and constraints of affinity propagation: • **Leader factors** ensure that each node selects exactly one cluster center.

$$\mathrm{f}_{\mathcal{E}^+(i,\cdot)}(\underline{x}_{\mathcal{E}(i,\cdot)}) = 1((\sum_{(i,j)\in\mathcal{E}^+(i,\cdot)} x_{i:j}) = 1) \quad \forall i \in \mathcal{V}$$

where as before $\mathcal{E}^+(i,.) = \{(i,j) \in \mathcal{E}\} \cup \{(i,i)\}$ is the set of edges leaving node i.

]indexfactor!consistency • **Consistency factors** as defined by equation (3.8) ensure that if any node $i \in \mathcal{E}(.,j)$ selects node j as its center of cluster, node j also selects itself as the center

At this point we note that we used both of these factor-types for set-cover and dominating set problem in section 3.5. The only addition (except for using a different semiring) is the following factors.

• **Local factors** take distances and the willingness to become a center into account

$$\mathrm{f}_{i:j}(x_{(i,j)}) = \min\left(\mathbf{A}_{i,j}, 1(x_{i:j} = 0)\right) \quad \forall (i,j) \in \mathcal{E} \cup \{(i,i) \mid i \in \mathcal{V}\}$$

where in effect $\mathrm{f}_{i:j}(x_{i:j})$ is equal to $\mathbf{A}_{i,j}$ if $x_{i:j} = 1$ and 0 otherwise

See equation (1.20) for definition of 1(.) in min-sum semiring and note the fact that $\oplus = \min$ in this case. This means extensions to other semirings (*e.g.*, min-max) need only to consider a different 1(.) and use their own $\oplus$ operator. However, direct application of min-max inference to this factor-graph is problematic for another reason: since the number of clusters K is not enforced, as soon as node i becomes center of a cluster, all nodes j with $\mathbf{A}_{j,j} \leq \mathbf{A}_{i,i}$ can become their own centers without increasing the min-max value. In section 4.4, we resolve this issue by enforcing the number of clusters K and use inference on min-max semiring to solve the corresponding problem known as K-center problem.

The complexity of min-sum BP with variable and factor synchronous message update is $O(|\mathcal{E}|)$ as each leader and consistency factor allows efficient $O(|\mathcal{E}(.,j)|)$ and $O(|\mathcal{E}(i,.)|)$ calculation of factor-to-variable messages respectively. Moreover, all variable-to-factor messages leaving node i can be calculated simultaneously in $O(|\mathcal{E}(i,.)|)$ using variable-synchronous update of equation (2.13).

4.1.1 Facility location problem

A closely related problem to K-medians is the facility location problem, where a matrix $\mathbf{A} \in \mathbb{R}^{|\mathcal{V}_1|\times|\mathcal{V}_2|}$ specifies the pairwise distance between two parts of bipartite graph $\mathcal{G} = (\mathcal{V} = (\mathcal{V}_1, \mathcal{V}_2), \mathcal{E})$.

The goal is to select a sub-set of facilities $\mathcal{D} \subset \mathcal{V}_1$ to minimize the sum of distances of each customer $i \in \mathcal{V}_2$ to its associated facility in $\mathcal{D}$.

The uncapacitated version of this problem (no restriction on $|\mathcal{D}|$) can be solved as special K-median problem where $\mathbf{A}_{i,i} = \infty \; \forall i \in \mathcal{V}_2$. Message passing solution for min-sum variations of this problem are discussed in [187, 188]. We discuss the min-max facility location as a special case of k-center problem in section 4.4.

4.2 Hierarchical clustering

By adding a dummy node $*$ and connecting all the cluster centers to this node with the cost $\mathbf{A}_{i,*} = \mathbf{A}_{i,i}$, we can think of the K-median clustering of previous section as finding a minimum cost tree of depth two with the dummy node as its root. The **shallow trees** of Bailly-Bechet et al. [18] generalize this notion by allowing more levels of hierarchy. The objective is to find a tree of maximum depth d, that minimizes the sum over all its edges. An alternative approach to hierarchical clustering based on nested application of affinity propagation is discussed in [111, 289].

Previously we presented the binary-variable model for affinity propagation. However, it is possible to obtain identical message updates using categorical variables [102]. Here $\underline{x} = \{x_1, \ldots, x_N\}$, where $x_i \in \mathcal{X}_i = \{j \mid (i,j) \in \mathcal{E}\}$ selects one of the neighbouring nodes as the center of the cluster for node i. In this case we can ignore the leader factors and change the consistency and local factors accordingly.

The idea in building hierarchies is to allow each node i to follow another node j even if i itself is followed by a third node k (*i.e.*, dropping the consistency factors). Also $x_i = i$ is forbidden. However, this creates the risk of forming loops. The trick used by Bailly-Bechet et al. [18] is to add an auxiliary "depth" variable $z_i \in \{0, \ldots, d\}$ at each node. Here, the depth of the dummy node $*$ is zero and the **depth factors** ensure that if i follows j then $z_i = z_j + 1$:

$$\mathsf{f}_{i,j}(\underline{x}_i, \underline{z}_{i,j}) \;=\; 1\Big(z_i = z_j + 1 \vee x_i \neq j\Big)$$

4.2.1 Spanning and Steiner trees

Although finding trees is not a clustering problem, since one could use the same techniques used for clustering by shallow trees, we include it in this section. Given a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, a penalty $w_{i:j} < 0$ per edge $(i,j) \in \mathcal{E}$ and a prize $w_i > 0$ per node $i \in \mathcal{V}$, the **prize-collecting Steiner tree**'s objective is to select a connected sub-graph $\mathcal{G}' = (\mathcal{V}' \subseteq \mathcal{V}, \mathcal{E}' \subseteq \mathcal{E})$ with maximum sum of prizes $\sum_{i \in \mathcal{V}'} w_i + \sum_{(i,j) \in \mathcal{E}'} w_{i:j}$. Since the optimal sub-graph is always a tree, a construction similar to that of shallow trees (with min-sum BP) finds high-quality solutions to depth-limited versions of this problem in $O(d|\mathcal{E}|)$ [19, 20, 23, 31].

The main difference with factor-graph of shallow trees is that here the root node is a pre-determined member of $\mathcal{V}$ and several tricks are used to find best (set of) root(s). However, since the $|\mathcal{V}'|$ may be smaller than $|\mathcal{V}|$, a different dummy node $*$ is introduced, with zero cost of connection to all nodes $i \in \mathcal{V}$ s.t. $x_i = *$ means node i is not a part of the Steiner tree ($i \in \mathcal{V} \setminus \mathcal{V}'$).

Alternatively, if we do not introduce this dummy node such that $\mathcal{V}' = \mathcal{V}$ and moreover, set the node penalties to zero, the result is a **depth limited spanning tree**. Bayati et al. [24] show that if the maximum depth is large enough (*e.g.*, $d = N$) and BP is convergent, it will find the minimum spanning tree.

4.3 K-clustering problem

Given a symmetric matrix of pairwise distances $\mathbf{A} \in \mathbb{R}^{N \times N}$ between N data-points, and a number of clusters K, K-clustering (a.k.a. **min-max clustering**) seeks a partitioning of data-points that minimizes the maximum distance between all the pairs in the same partition.

We formulate this problem as **min-max** inference problem in a factor-graph. Let $\underline{x} = \{x_1, \ldots, x_N\}$ with $x_i \in \{1, \ldots, K\}$ be the set of variables, where $x_i = k$ means, point i belongs to cluster k. The Potts factor

$$\mathsf{f}_{\{i,j\}}(x_i, x_j) = \min\left(\mathbb{1}(x_i \neq x_j), \mathbf{A}_{i,j}\right) \quad \forall 1 \leq i < j \leq N \tag{4.1}$$

is equal to $\mathbf{A}_{i,j}$ if two nodes are in the same cluster and $-\infty$ otherwise. It is easy to see that using min-max inference on this factor graph, the min-max solution $\underline{x}^*$ (equation (1.26)) defines a clustering that minimizes the maximum of all inter-cluster distances.

Recall that the y-neighborhood graph (definition 3.6.1) for a distance matrix $\mathbf{A}$ is the graph $\mathcal{G}(\mathbf{A}, y) = (\mathcal{V}, \mathcal{E}(\mathbf{A}, y) = \{(i,j) \mid \mathbf{A}_{i,j} \leq y\})$.

Claim 4.3.1. *The p_y-reduction of the min-max clustering factor-graph above is identical to K-clique-cover factor-graph of section 3.4 for $\mathcal{G}(\mathbf{A}, y)$.*

Proof. The p_y-reduction of the K-clustering factor (equation (4.1)) is

$$\mathsf{f}_{\{i,j\}}(x_i, x_j) = \mathbb{1}\left(\min\left(\mathbb{1}(x_i \neq x_j), \mathbf{A}_{i,j}\right) \leq y\right) = \mathbb{1}(x_i \neq x_j \vee \mathbf{A}_{i,j} \leq y)$$

Recall that the K-clique-cover factor-graph, defines a pairwise factor between any two nodes i and j whenever $(i,j) \notin \mathcal{E}(\mathbf{A}, y)$, – *i.e.*, whenever $\mathbf{A}_{i,j} \leq y$ it does not define a factor. However, $\mathbf{A}_{i,j} \leq y$ means that the reduced constraint factor above is satisfied and therefore we only need to consider the cases where $\mathbf{A}_{i,j} > y$. This gives $\mathsf{f}_{\{i,j\}}(x_i, x_j) = \mathbb{1}(x_i \neq x_j)$, which is the K-clique-cover factor between two nodes i and j s.t. $(i,j) \notin \mathcal{E}(\mathbf{A}, y)$. □

We use binary-search over sum-product reductions to approximate the min-max solution (see section 1.4.2). Considering the $O(N^2K)$ cost of message passing for K-clique-cover, this gives $O(N^2K\log(N^2)) = O(N^2K\log(N))$ cost for K-clustering, where the $\log(N^2)$ is the cost of binary search over the set of all possible pairwise distance values in $\mathbf{A}$.

Figure 4.1 compares the performance of min-max clustering using message passing to that of Furthest Point Clustering (FPC) [119] which is 2-optimal when the triangle inequality holds. Note that message passing solutions are superior even when using Euclidean distance.

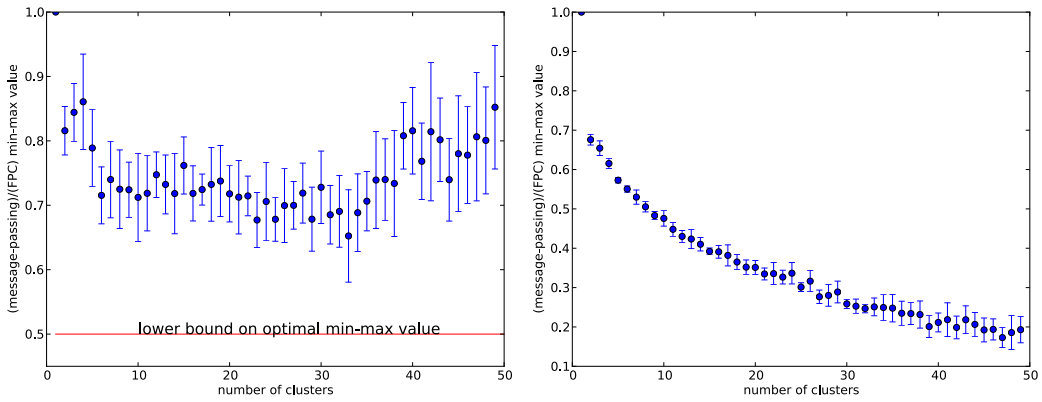


Figure 4.1: *Min-max clustering of 100 points with varying numbers of clusters (x-axis). Each point is an average over 10 random instances. The y-axis is the ratio of the min-max value obtained by sum-product reduction (and using perturbed BP with $T = 50$ to find satisfying assignments) divided by the min-max value of Furthest Point Clustering (FPC).* ***(left)*** *Clustering of random points in 2D Euclidean space. The red line is the lower bound on the optimal result based on the worst case guarantee of FPC.* ***(right)*** *Using symmetric random distance matrix where $\mathbf{A}_{i,j} = \mathbf{A}_{j,i} \sim U(0,1)$.*

4.4 K-center problem

Given a pairwise distance matrix $D \in \mathbb{R}^{N\times N}$, the K-center problem seeks a partitioning of nodes, with one center per partition such that the maximum distance from any node to the center of its partition is minimized.

This problem is known to be $\mathbb{NP}$-hard, even for Euclidean distance matrices [201], however K-center has some approximation algorithms that apply when the distance matrix satisfies the triangle inequality [87, 135]. The method of Dyer and Frieze [87] is very similar to furthest point clustering and extends to weighted K-center problem in which the distance from any point to all the other points is scaled by its weight. The more general case of asymmetric distances does not allow any constant-factor approximation (however $o(\log(N))$-approximation exists [66, 237]).

Here we define a factor-graph, whose min-max inference results in the optimal solution for K-center problem. For this consider the graph $\mathcal{G}(\mathcal{V},\mathcal{E})$ induced by the distance matrix $\mathbf{A}$ s.t. $\mathcal{V} = \{1,\dots,N\}$ and $\mathbf{A}_{i,j} = \infty \Leftrightarrow (i,j) \notin \mathcal{E}$.

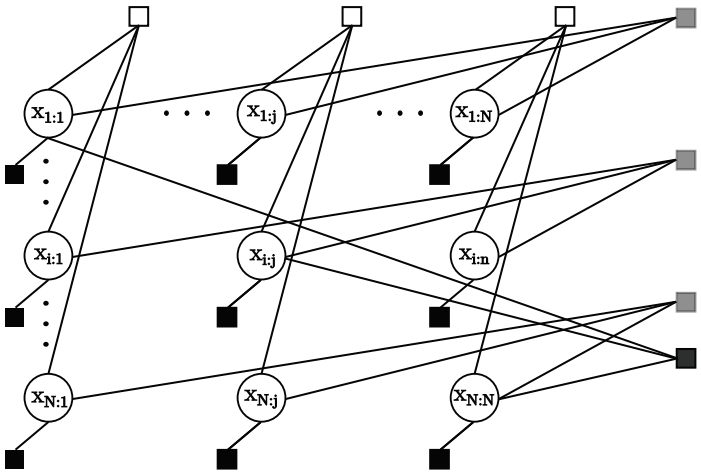


Figure 4.2: *The factor-graph for K-center problem, where the local factors are black squares, the leader factors are light grey, consistency factors are white and K-of-N factor is dark grey.*

Let $\underline{x} = \{x_{i:j} \mid (i,j) \in \mathcal{E}\}$ be the set of variables, where $x_{i:j} \in \{0,1\}$ indicates whether j is the center of cluster for i. Define the following factors:

• **Local factors:**

$$\mathrm{f}_{(i,j)}(x_{i:j}) = \min\left(\mathbf{A}_{i,j}, 1(x_{i:j} = 0)\right) \quad \forall (i,j) \in \mathcal{E}$$

• **Leader, Consistency and at-most-K-of-N factors** as defined for the sum-product case of induced K-set-cover and K-independent set (section 3.5). Here we need to replace sum-product $1(.)$ with the min-max version (equation (1.20)).

For variants of this problem such as the capacitated K-center, additional constraints on the maximum/minimum points in each group may be added as the at-least/at-most K-of-N factors.

Claim 4.4.1. *The* p_y*-reduction of the K-center clustering factor-graph above is identical to K-set-cover factor-graph of section 3.5 for* $\mathcal{G}(\mathbf{A}, y)$.

Proof. Leader, consistency and at-most-K-of-N factors are identical to the factors used for K-set-cover (which is identical to their p_y-reduction), where the only difference is that here we have one variable per each $(i,j) \in \mathcal{E}$ whereas in K-set-cover for $\mathcal{G}(\mathbf{A}, y)$ we have one variable per $(i,j) \in \mathcal{E} \mid \mathbf{A}_{i,j} \leq y$. By considering the p_y-reduction of the local factors

$$\mathrm{f}_{i:j}(x_{i:j}) = 1\left(\min\left(\mathbf{A}_{i,j}, 1(x_{i:j)} = 0)\right) \leq y\right) = 1(\mathbf{A}_{i,j} \leq y \vee x_{i:j} = 0)$$

we see that we can assume that for $\mathbf{A}ij > 0$, $x_{i:j} = 0$ and we can drop these variables from the factor-graph. After this we can also omit the p_y-reduction of the local factors as they have no effect. This gives us the factor-graph of section 3.5 for set-cover. □

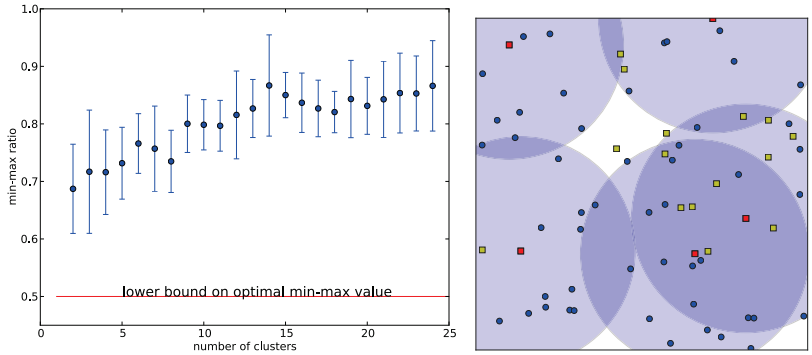


Figure 4.3: (left) *K-center clustering of 50 random points in a 2D plane with various numbers of clusters (x-axis). The y-axis is the ratio of the min-max value obtained by sum-product reduction ($T = 500$ for perturbed BP) over the min-max value of 2-approximation of [87].* ***(right)*** *Min-max K-facility location formulated as an asymmetric K-center problem and solved using message passing. Yellow squares indicate 20 potential facility locations and small blue circles indicate 50 customers. The task is to select 5 facilities (red squares) to minimize the maximum distance from any customer to a facility. The radius of circles is the min-max value.*

Similar to the K-clustering problem, we use the sum-product reduction to find near-optimal solutions to this problem. The binary search seeks the optimal $y \in \mathcal{Y}$ (where $\mathcal{Y}$ is the collective range of all the factors). Here, since only local factors take values other than $\pm\infty$, the search is over their range, which is basically all the values in $\mathbf{A}$. This adds an additional $\log(N)$ multiplicative factor to the complexity of K-set-cover (which depends on the message update).

We can significantly reduce the number of variables and the complexity by bounding the distance to the center of the cluster $\overline{y}$. Given an upper bound $\overline{y}$, we may remove all the variables $x_{i:j}$ where $\mathbf{A}_{i,j} > \overline{y}$ from the factor-graph. Assuming that at most R nodes are at distance $\mathbf{A}_{i,j} \leq \overline{y}$ from every node j, the complexity of min-max inference with synchronous update drops to $O(NR^2 \log(N))$. This upper bound can be obtained for example by applying approximation algorithms.

Figure 4.3(left) compares the performance of message-passing and the 2-approximation of [87] when triangle inequality holds. The min-max facility location problem can also be formulated as an asymmetric K-center problem where the distance *to* all customers is ∞ and the distance from a facility to another facility is $-\infty$ (figure 4.3(right)).

4.5 Modularity maximization

A widely used objective for clustering (or community mining) is Modularity maximization [229]. However, exact optimization of Modularity is $\mathbb{NP}$-hard [45]. Modularity is closely related to fully connected Potts graphical models [259, 332]. Many have proposed various other heuristics for modularity optimization [34, 67, 228, 259, 265]. Here after a brief review of the Potts model in section 4.5.1, we introduce a factor-graph representation of this problem that has a large number

of factors in section 4.5.2. We then use the augmentation technique (section 2.2.2) to incrementally incorporate violated constraints in these models.

4.5.1 The Potts model

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ be an undirected graph, with $M = |\mathcal{E}|$, $N = |\mathcal{V}|$ and the adjacency matrix $\mathbf{B} \in \mathbb{R}^{N \times N}$, where $\mathbf{B}_{i,j} \neq 0 \Leftrightarrow (i,j) \in \mathcal{E}$. Let $\mathbf{A}$ be the normalized adjacency matrix where $\sum_{i<j} \mathbf{A} = 1$. Also let $\mathbf{A}_{\mathcal{E}(i,\cdot)} \stackrel{\text{def}}{=} \sum_j \mathbf{A}_{i,j}$ denote the normalized degree of node v_i. Graph clustering using **modularity** optimization seeks a partitioning of the nodes into unspecified number of clusters $C = \{C_1, \ldots, C_K\}$, maximizing

$$\mathsf{m}(C) \stackrel{\text{def}}{=} \sum_{C_i \in C} \sum_{i,j \in C_i} \left(\mathbf{A}_{i,j} - \zeta \mathbf{A}_{\mathcal{E}(i,\cdot)} \mathbf{A}_{\mathcal{E}(j,\cdot)} \right) \tag{4.2}$$

The first term of modularity is proportional to within-cluster edge-weights. The second term is proportional to the expected number of within cluster edge-weights for a null model with the same weighted node degrees for each node i. Here the null model is a fully-connected graph. The **resolution** parameter ζ – which is by default set to one – influences the size of communities, where higher resolutions motivates a larger number of clusters.

In the Potts model, each node $i \in \mathcal{V}$ is associated with a variable $x_i \in \{1, \ldots, K_{\max}\}$, where $K_{\max}$ is an upper bound on the number of clusters. Here, each pair of variables have a pairwise interaction

$$\mathsf{f}_{\{i,j\}}(\underline{x}_{\{i,j\}}) = \begin{cases} \zeta(\mathbf{A}_{\mathcal{E}(i,\cdot)} \mathbf{A}_{\mathcal{E}(j,\cdot)}) - \mathbf{A}_{i,j} & x_i = x_j \\ 0 & x_i \neq x_j \end{cases} \tag{4.3}$$

where min-sum inference on this fully connected factor-graph gives an assignment of each node to a cluster so as to maximize the modularity.

4.5.2 Clique model

Here, we introduce an alternative factor-graph for modularity optimization. Before introducing our factor-graph representation, we suggest a procedure to stochastically approximate the null model using a sparse set of interactions.

We generate a random *sparse null model* with $M^{\text{NULL}} <_{\alpha\!\!\!/} M$ weighted edges ($\mathcal{E}^{\text{NULL}}$), by randomly sampling two nodes, each drawn independently from $\Pr(i) \propto \sqrt{\mathbf{A}_{\mathcal{E}(i,\cdot)}}$, and connecting them with a weight proportional to $\mathbf{B}^{\text{NULL}}_{i,j} \propto \sqrt{\mathbf{A}_{\mathcal{E}(i,\cdot)} \mathbf{A}_{\mathcal{E}(j,\cdot)}}$. If they have been already connected, this weight is added to their current weight. We repeat this process $\alpha\!\!\!/ M$ times, however since some of the edges are repeated, the total number of edges in the sparse null model may be less than $\alpha\!\!\!/ M$.

Finally the normalized edge-weight in the sparse null model is

$$\mathbf{A}^{\text{NULL}}_{i,j} \stackrel{\text{def}}{=} \frac{\mathbf{B}^{\text{NULL}}_{i,j}}{\sum_{k,l} \mathbf{B}^{\text{NULL}}_{k,l}}.$$

It is easy to see that this generative process in expectation produces the fully connected null model.[1]

Factor-graph representation

Here we use the following binary-valued factor-graph formulation. Let $\underline{x} = \{x_{i:j} \in \{0,1\} \mid (i,j) \in \mathcal{E} \cup \mathcal{E}^{\text{NULL}}\}$ be a set of binary variables, and let L denote the cardinality of $\mathcal{E} \cup \mathcal{E}^{\text{NULL}}$. The variable $x_{i:j}$ is equal to one means the corresponding edge, (i,j), is present in the final model, where our goal is to define the factor-graph such that the final model consists of cliques. For this, define the factors as follows:

• *Local factor* for each variable are equal to the difference between the weighted adjacency and the null model if an edge is present (*i.e.*, $x_{i:j} = 1$)

$$\mathsf{f}_{i:j}(x_{i:j}) = \min\left(1(x_{i:j} = 0), \mathbf{A}^{\text{NULL}}_{i,j} - \mathbf{A}_{i,j}\right) \tag{4.4}$$

By enforcing the formation of cliques, while minimizing the sum of local factors the negative sum of local factors evaluates to modularity (equation (4.2)):

• For each three edges $(i,j),(j,k),(i,k) \in \mathcal{E} \cup \mathcal{E}^{\text{NULL}}, i < j < k$ that form a triangle, define a **clique factor** as

$$\mathsf{f}_{\{i:j,j:k,i:k\}}(x_{i:j}, x_{j:k}, x_{i:k}) = 1(x_{i:j} + x_{j:k} + x_{i:k} \neq 2) \tag{4.5}$$

These factors ensure the formation of cliques – *i.e.*, if two edges that are adjacent to the same node are present (*i.e.*, $x_{i:i} = 1$ and $x_{i:k} = 1$), the third edge in the triangle should also be present (*i.e.*, $x_{j:k} = 1$). The computational challenge here is the large number of clique constraints. For the fully connected null model, we need $O(N^3)$ such factors and even using the sparse null model – assuming a random edge probability a.k.a. Erdos-Reny graph – there are $O(\frac{L^3}{N^6}N^3) = O(\frac{L^3}{N^3})$ triangles in the graph (recall that $L = |\mathcal{E} \cup \mathcal{E}^{\text{NULL}}|$).

Brandes *et al.* [45] first introduced an LP formulation with similar form of constraints. However, since they include all the constraints from the beginning and the null model is fully connected, their method is only applied to small toy problems.

[1] The choice of using square root of weighted degrees for both sampling and weighting is to reduce the variance. One may also use pure importance sampling (*i.e.*, use the product of weighted degrees for sampling and set the edge-weights in the null model uniformly), or uniform sampling of edges, where the edge-weights of the null model are set to the product of weighted degrees.

Simplified message update and augmentation

Here, we give technical details as how to simplify the min-sum BP message update for this factor-graph. The clique factor is satisfied only if either zero, one, or all three of the variables in its domain are non-zero. Therefore, in order to derive message updates $\widehat{\mathsf{p}}_{\{i:j,j:k,i:k\}\to i:j}$ from the clique factor $\mathsf{f}_{\{i:j,j:k,i:k\}}$ to variable $x_{i:j}$ for a particular value of $x_{i:j}$ (*e.g.*, $x_{i:j} = 0$), we apply $\oplus$ operator (*i.e.*, minimize) over all the valid cases of incoming messages (*e.g.*, when $x_{i:k}$ and $x_{j:k}$ in clique factor $\mathsf{f}_{\{i:j,j:k,i:k\}}$ are zero). This gives the simplified factor-to-variable messages

$$\begin{aligned}\widehat{\mathsf{p}}_{\{i:j,j:k,i:k\}\to i:j}(0) &= \min\{0,\ \widehat{\mathsf{p}}_{j:k\to\{i:j,j:k,i:k\}},\ \widehat{\mathsf{p}}_{i:k\to\{i:j,j:k,i:k\}}\}\\ \widehat{\mathsf{p}}_{\{i:j,j:k,i:k\}\to i:j}(1) &= \min\{0,\ \widehat{\mathsf{p}}_{j:k\to\{i:j,j:k,i:k\}} \ +\ \widehat{\mathsf{p}}_{i:k\to\{i:j,j:k,i:k\}}\}\end{aligned} \tag{4.6}$$

where for $x_{i:j} = 0$, the minimization is over three feasible cases (a) $x_{j:k} = x_{i:k} = 0$, (b) $x_{j:k} = 1, x_{i:k} = 0$ and (c) $x_{j:k} = 0, x_{i:k} = 1$. For $x_{i:j} = 1$, there are two feasible cases (a) $x_{j:k} = x_{i:k} = 0$ and (b) $x_{j:k} = x_{i:k} = 1$.

Here we work with normalized messages, such that $\widehat{\mathsf{p}}_{\mathrm{I}\to i}(0) = 0$ [2] and use $\widehat{\mathsf{p}}_{\mathrm{I}\to i}$ to denote $\widehat{\mathsf{p}}_{\mathrm{I}\to i}(1)$. The same applies to the marginal $\widehat{\mathsf{p}}_{i:j}$, which is a scalar called bias. Here $\widehat{\mathsf{p}}_{i:j} > 0$ means $\widehat{\mathsf{p}}_{i:j}(1) > \widehat{\mathsf{p}}_{i:j}(0)$ and shows a preference for $x_{i:j} = 1$. Normalizing clique-factor messages above we get the following form of simplified factor-to-variable messages for clique constraints

$$\begin{aligned}\widehat{\mathsf{p}}_{\{i:j,j:k,i:k\}\to i:j} = \min\{0,\ \widehat{\mathsf{p}}_{j:k\to\{i:j,j:k,i:k\}} \ +\ \widehat{\mathsf{p}}_{i:k\to\{i:j,j:k,i:k\}}\} - \\ \min\{0,\ \widehat{\mathsf{p}}_{j:k\to\{i:j,j:k,i:k\}},\ \widehat{\mathsf{p}}_{i:k\to\{i:j,j:k,i:k\}}\}.\end{aligned} \tag{4.7}$$

In order to deal with large number of factors in this factor-graph, we use the augmentation approach of section 2.2.2. We start with no clique-factors, and run min-sum BP to obtain a solution (which may even be unfeasible). We then find a set of clique constraints that are violated in the current solution and augment the factor-graph with factors to enforce these constraints. In order to find violated constraints in the current solution, we simply look at pairs of positively fixed edges ($x_{i:j} = 1$ and $x_{i:k} = 1$) around each node i and if the third edge of the triangle is not positively fixed ($x_{j:k} = 0$), we add the corresponding clique factor ($\mathsf{f}_{\{i:j,j:k,i:k\}}$) to the factor-graph. See algorithm 3 (in the appendix) for details of our message-passing for Modularity maximization.

Experiments

We experimented with a set of classic benchmarks[3]. Since the optimization criteria is modularity, we compared our method only against best known "modularity optimization" heuristics:

[2] Note that this is different from the standard normalization for min-sum semiring in which $\min_{x_i} \widehat{\mathsf{p}}_{\mathrm{I}\to i}(x_i) = 0$.

[3] Obtained form Mark Newman's website: http://www-personal.umich.edu/~mejn/netdata/

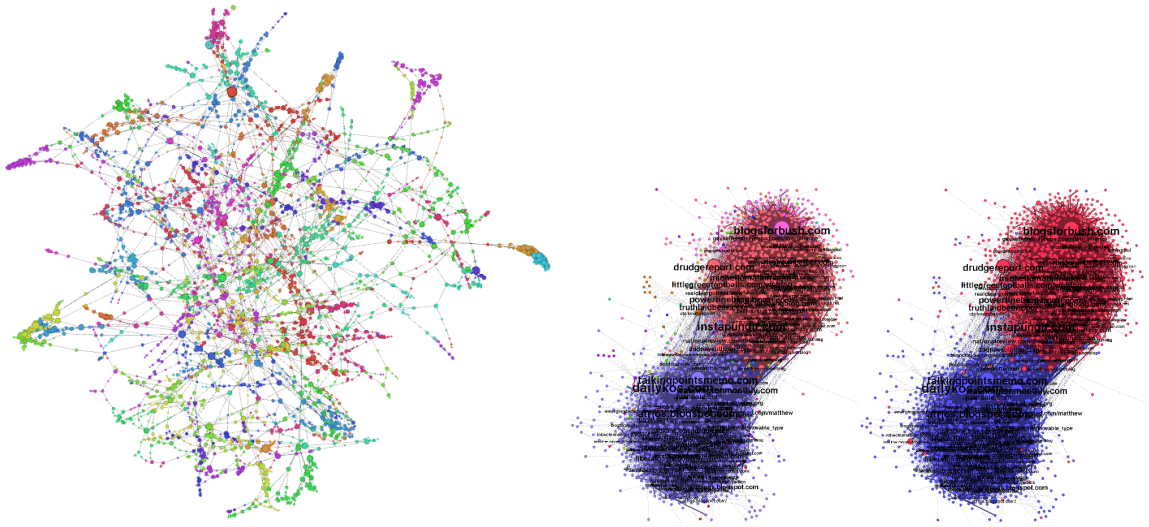


Figure 4.4: *(left) Clustering of power network ($N = 4941$) by message passing. Different clusters have different colors and the nodes are scaled by their degree. (right) Clustering of politician blogs network ($N = 1490$) by message passing and by meta-data* – i.e., *liberal or conservative.*

Table 4.1: *Comparison of different modularity optimization methods.*

				message passing (full)				message passing (sparse)				Spin-glass		L-Eigenvector		FastGreedy		Louvian	
Problem	Weighted?	Nodes	Edges	L	Cost	Modularity	Time	L	Cost	Modularity	Time	Modularity	Time	Modularity	Time	Modularity	Time	Modularity	Time
polbooks	y	105	441	5461	5.68%	0.511	.07	3624	13.55%	0.506	.04	0.525	1.648	0.467	0.179	0.501	0.643	0.489	0.03
football	y	115	615	6554	27.85%	0.591	0.41	5635	17.12%	0.594	0.14	0.601	0.87	0.487	0.151	0.548	0.08	0.602	0.019
wkarate	n	34	78	562	12.34%	0.431	0	431	15.14%	0.401	0	0.444	0.557	0.421	0.095	0.410	0.085	0.443	0.027
netscience	n	1589	2742	NA	NA	NA	NA	53027	**.0004**%	0.941	2.01	0.907	8.459	0.889	0.303	0.926	0.154	0.948	0.218
dolphins	y	62	159	1892	14.02%	0.508	0.01	1269	6.50%	0.521	0.01	0.523	0.728	0.491	0.109	0.495	0.107	0.517	0.011
lesmis	n	77	254	2927	5.14%	0.531	0	1601	1.7%	0.534	0.01	0.529	1.31	0.483	0.081	0.472	0.073	0.566	0.011
celegansneural	n	297	2359	43957	16.70%	0.391	10.89	21380	3.16%	0.404	2.82	0.406	5.849	0.278	0.188	0.367	0.12	0.435	0.031
polblogs	y	1490	19090	NA	NA	NA	NA	156753	**.14**%	0.411	32.75	0.427	67.674	0.425	0.33	0.427	0.305	0.426	0.099
karate	y	34	78	562	14.32%	0.355	0	423	17.54%	0.390	0	0.417	0.531	0.393	0.086	0.380	0.079	0.395	0.009

(a) FastModularity[67], (b) Louvain [34], (c) Spin-glass [259] and (d) Leading eigenvector [228]. [4]

table 4.1 summarizes our results (see also Figure 4.4). Here for each method and each data-set, we report the *time* (in seconds) and the *Modularity* of the communities found by each method. The table include the results of message passing for both full and sparse null models, where we used a constant $\alpha = 20$ to generate our stochastic sparse null model. For message passing, we also included $L = |\mathcal{E} + \mathcal{E}^{\text{NULL}}|$ and the saving in the *cost* using augmentation. This column shows the percentage of the number of all the constraints considered by the augmentation. For example, the cost of $.14\%$ for the polblogs data-set shows that augmentation and sparse null model meant using $.0014$ times fewer clique-factors, compared to the full factor-graph.

Overall, the results suggest that our method is comparable to state-of-the-art in terms both time and quality of clustering. Although, we should note that the number of triangle constraints in large and dense graphs increases very quickly, which deteriorates the performance of this approach despite using the augmentation. Despite this fact, our results confirm the utility of augmentation by showing that it is able to find feasible solutions using a very small portion of the constraints.

[4] For message passing, we use $\lambda = .1$, $\epsilon_{\max} = \text{median}\{|\mathbf{A}_{i,j} - \mathbf{A}^{\text{NULL}}_{i,j}|\}_{(i,j)\in\mathcal{E}\cup\mathcal{E}^{\text{NULL}}}$ and $T_{\max} = 10$. Here we do not perform any decimation and directly fix the variables based on their bias $\widehat{\rho}_{i:j} > 0 \Leftrightarrow x_{i:j} = 1$.

Chapter 5

Permutations

5.1 Matching and permanent

The integration and maximization problems over unrestricted permutations define several important combinatorial problems. Two notable examples of integration problems are **permanent and determinant** of a matrix. Determinant of matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ is defined as

$$\det(\mathbf{A}) = \sum_{\underline{x} \in \mathcal{S}_N} \text{sign}(\underline{x}) \prod_{i=1}^{N} \mathbf{A}_{i,x_i}$$

where $\mathcal{S}_N$ is the set of all permutations of N elements (a.k.a. symmetric group) and $x_i \in \{1, \ldots, N\}$ is the index of i^{th} element in particular permutation $\underline{x}$. Here the sign(.) classifies permutations as even ($\text{sign}(\underline{x}) = 1$) and odd ($\text{sign}(\underline{x}) = -1$), where we can perform an even (odd) permutation by even (odd) number of pairwise exchanges. The only difference in definition of permanent is removal of the sign function

$$\text{perm}(\mathbf{A}) = \sum_{\underline{x} \in \mathcal{S}_N} \prod_{i=1}^{N} \mathbf{A}_{i,x_i}$$

Here, we see that both permanent and determinant are closely related with two easy combinatorial problems on graphs – *i.e.*, perfect matching and spanning sub-tree. While calculating the permanent for $\mathbf{A} \in \{0,1\}^{N \times N}$ is $\#\mathbb{P}$-hard [297], the determinant can be obtained in $O(N^3)$ [116].

The **matrix-tree theorem** states that the number of spanning trees in a graph with adjacency matrix $\mathbf{A}$ is equal to $\det(\mathbf{L}(i,i))$ for an arbitrary $1 \leq i \leq N$. Here $\mathbf{L} = \mathbf{A} - \mathbf{D}$ is the Laplacian of $\mathbf{A}$, where $\mathbf{D}$ is a diagonal matrix with degree of each node on the diagonal (*i.e.*, $\mathbf{D}_{i,i} = \sum_j \mathbf{A}_{i,j}$) and $\mathbf{L}(i,i)$ is the $(N-1) \times (N-1)$ sub-matrix of $\mathbf{L}$ in which row and column i are removed.

An intermediate step in representing the permutation as a graphical model is to use a **bipartite**

graph $\mathcal{G} = (\mathcal{V} = (\mathcal{V}_1, \mathcal{V}_2), \mathcal{E})$, where $|\mathcal{V}_1| = |\mathcal{V}_2| = N$ and the edge-set $\mathcal{E} = \{(i,j) \mid i \in \mathcal{V}_1, j \in \mathcal{V}_2\}$. A **perfect matching** is a one to one mapping of elements of $\mathcal{V}_1$ to $\mathcal{V}_2$ and can be represented using the corresponding edges $\mathcal{E}' \subset \mathcal{E}$. It is easy to see that any perfect matching $\mathcal{E}' \subset \mathcal{E}$ identifies a permutation $\underline{x}$. Here the maximum weighted matching (a.k.a. assignment problem) problem is to find a perfect matching $\underline{x}^* = \arg_{\underline{x} \in \mathcal{S}_N} \max \prod_{i=1}^N \mathbf{A}_{i,x_i}$, while the **bottleneck** assignment problem seeks $\underline{x}^* = \arg_{\underline{x} \in \mathcal{S}_N} \min \max_{i=1}^N \mathbf{A}_{i,x_i}$. The factor-graph representation of the next section shows that bipartite matching and bottleneck assignment problems correspond to the max-product (min-sum) and min-max inference, and computation of permanent corresponds to sum-product inference over the same factor-graph.

Interestingly min-sum and min-max inference in this setting are in $\mathbb{P}$ [121, 182] while sum-product is in $\#\mathbb{P}$. Indeed the application of max-product BP to find the maximum weighted matching [22] (and its generalization to maximum weighted b-matching [138]) is one of the few cases in which loopy BP is guaranteed to be optimal. Although MCMC methods (section 2.6.1) can provide polynomial time approximation schemes for permanent [150] (and many other combinatorial integration problems [149]), they are found to be slow in practice [139]. This has motivated approximations using deterministic variational techniques [62, 308] and in particular BP [139], which is guaranteed to provide a lower bound on the permanent [301].

5.1.1 Factor-graph and complexity

Here we review the factor-graph of Bayati et al. [22] for maximum bipartite matching. Given the bipartite graph $\mathcal{G} = ((\mathcal{V}_1, \mathcal{V}_2), \mathcal{E})$ and the associated matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$ with non-negative entries, define two sets of variables $\underline{x} = \{x_i \in \mathcal{V}_2 \mid i \in \mathcal{V}_1\}$ and $\underline{z} = \{z_j \in \mathcal{V}_1 \mid j \in \mathcal{V}_2\}$, where $x_i = j$ and $z_j = i$ both mean node i is connected to node j. Obviously this representation is redundant and for $(i,j) \in \mathcal{E}$ a pairwise factor should ensure x_i and z_i are consistent:

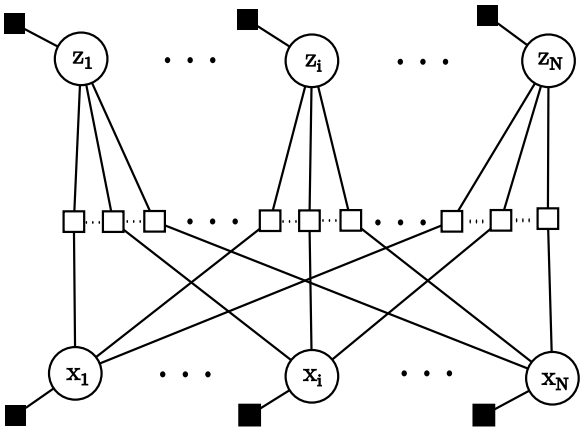


Figure 5.1: *The factor-graph for matching where the local factors are black and consistency factors are white squares.*

• **Consistency factors** ensure the consistency of $\underline{x}$ and $\underline{z}$

$$\mathsf{f}_{\{i,j\}}(x_i, z_j) \;=\; 1\Big((x_i = j \wedge z_j = i) \vee (x_i \neq j \wedge z_j \neq i)\Big) \quad \forall (i,j) \in \mathcal{E}$$

• **Local factors** represent the cost of a matching

$$\mathsf{f}_{\{i\}}(x_i) \;=\; \mathbf{A}_{i,x_i} \qquad\qquad \forall i \in \mathcal{V}_1$$

where if i and j are connected in a matching, two local factors $\mathsf{f}_{\{i\}}(x_j) = \sqrt{\mathbf{A}_{i,j}}$ and $\mathsf{f}_{\{j\}}(x_i) = \sqrt{\mathbf{A}_{i,j}}$ account for $\mathbf{A}_{i,j}$. Figure 5.1 shows this factor-graph.

It is easy to see that the joint form $\mathsf{q}(\underline{x}, \underline{z})$ is equal to $\bigotimes_{i=1}^{N} \mathbf{A}_{i,x_i}$ for any consistent assignment to $\underline{x}, \underline{z}$ and it is equal to $\overset{\oplus}{1}$ otherwise. Therefore sum-product and max-product (*i.e.*, $\sum_{\underline{x},\underline{z}} \mathsf{q}(\underline{x}, \underline{z})$ and $\max_{\underline{x},\underline{z}} \mathsf{q}(\underline{x}, \underline{z})$) produce the permanent and max-weighted matching respectively.

The cost of each iteration of both max-product and sum-product BP in the factor-graph above is $O(N^2)$. Moreover, for max-product BP, if the optimal solution is unique, BP is guaranteed to converge to this solution after $O(\frac{Ny^*}{\epsilon})$ iterations, where ϵ is the difference between the cost of first and second best matchings and y^* is the cost of best matching [22].

An alternative is to use a **binary variable model** in which each edge of the bipartite graph is associated with a binary variable and replace the consistency factors with **degree constraint** to ensure that each node is matched to exactly one other node. However this model results in BP updates equivalent to the one above (the simplification of updates discussed in [25] is exactly the updates of binary variable model).

For min-max semiring, $\mathsf{q}(\underline{x}, \underline{z}) = \max_{i=1}^{N} \mathbf{A}_{i,x_i}$ for a consistent assignment to $\underline{x}, \underline{z}$ and it evaluates to $\overset{\min}{1} = \infty$. Therefore min-max inference here seeks an assignment that minimizes the maximum matching cost – a.k.a. **bottleneck assignment** problem.

5.1.2 Arbitrary graphs

We can also use message passing to solve max-weighted matching in an arbitrary graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with adjacency matrix $\mathbf{A}$. Zdeborová and Mézard [331] proposed a factor-graph for the related task of counting of the perfect matchings in an arbitrary graph. For this, each edge $(i,j) \in \mathcal{E}$ is assigned to one binary variable $x_{i:j} \in \{0,1\}$ and the *degree factors* on each node restrict the number of non-zero values to one

$$\mathsf{f}_{\mathcal{E}(i,.)}(\underline{x}_{\mathcal{E}(i,.)}) \quad = \quad 1\Big(\big(\sum_{(i,j)\in\mathcal{E}(i,.)} x_{i:j}\big) \leq 1\Big)$$

where $\mathcal{E}(i,.)$ is the set of all the edges adjacent to node i.

Sanghavi [273] consider the problem of maximum weighted b-matching in arbitrary graphs by changing the degree factor to

$$\mathsf{f}_{\mathcal{E}(i,.)}(\underline{x}_{\mathcal{E}(i,.)}) \quad = \quad 1((\sum_{(i,j)\in\mathcal{E}(i,.)} x_{i:j}) \leq b)$$

and also *local factor* $\mathsf{f}_{i:j}(x_{i:j}) = x_{i:j}\mathbf{A}_{i,j}$ that takes the weights into account. They show that if the solution to the corresponding LP relaxation is integral then BP converges to the optimal solution. Moreover, BP does not converge if the LP solution is not integral.

Matchings in an arbitrary graph $\mathcal{G}$ is also related to the permanent and also (vertex disjoint) **cycle covers**; a set of directed cycles in $\mathcal{G}$ that cover all of its nodes exactly once. The number of such cycle covers in an un-weighted graph is equal to $\mathsf{perm}(\mathbf{A})$, which is in turn equal to the square of number of perfect matchings [222, ch. 13].

In fact a directed cycle cover with maximum weight is equivalent to the maximum weighted bipartite matching in the construction of the previous section. In section 5.2 below we will use message passing to obtain a minimum weighted "undirected" cycle cover and further restrict these covers to obtain a minimum weighted cover with a single cycle – *i.e.*, a minimum tour for TSP.

5.2 Traveling salesman problem

A Traveling Salesman Problem (TSP) seeks the minimum length tour of N cities that visits each city exactly once. TSP is $\mathbb{NP}$-hard and for general distances, no constant factor approximation to this problem is possible [238]. The best known exact solver, due to Held and Karp [128], uses dynamic programming to reduce the cost of enumerating all orderings from $O(N!)$ to $O(N^2 2^N)$. The development of many (now) standard optimization techniques are closely linked with advances in solving TSP. Important examples are simulated annealing [56, 166], mixed integer linear programming [118], dynamic programming [28], ant colony optimization [85] and genetic algorithms [115, 120].

Since Dantzig et al. [77] manually applied the cutting plane method to 49-city problem, a combination of more sophisticated cuts, used with branch-and-bound techniques [21, 233] has produced the state-of-the-art TSP-solver, Concorde [13]. Other notable results on very large instances have been reported by Lin-Kernigan heuristic [130] that continuously improves a solution by exchanging nodes in the tour. For a readable historical background of the state-of-the-art in TSP and its applications, see [12].

The search over the optimal tour is a search over all permutations of N cities that contains no sub-tours – that is the permutation/tour is constrained such that we dot not return to the starting city without visiting all other cities. Producing the permutation with minimum cost that may include sub-tours is called the (vertex disjoint) **cycle-cover** and is in $\mathbb{P}$ (see section 5.1.2).

We provide two approaches to model TSP: section 5.2.1 presents the first approach, which ignores the subtour constraints – *i.e.*, finds cycle covers – and then "augment" the factor-graph with such constraints when they become violated. This augmentation process is repeated until a feasible solution is found. The second approach, presented in section 5.2.2, is to use the variables that represent the time-step in which a node is visited. By having the same number of time-steps as cities, the subtour constraint is automatically enforced. This second formulation, which is computationally more expensive, is closely related to our factor-graph for sub-graph isomorphism (see section 5.3). This is because one can think of the problem of finding a **Hamiltonian cycle** in $\mathcal{G}$ as finding a sub-graph of $\mathcal{G}$ that is isomorphic to a loop of size $|\mathcal{V}|$.

5.2.1 Augmentative approach

Let $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ denote a graph of our problem with the positively weighted symmetric adjacency matrix $\mathbf{A}$, s.t. $\mathbf{A}_{i,j} = 0 \Leftrightarrow (i,j) \notin \mathcal{E}$. The objective is to select a subset of $\mathcal{E}$ that identifies shortest tour of N cities. Let $\underline{x} = \{x_{i:j} \mid (i,j) \in \mathcal{E}\}$ be a set of M binary variables (*i.e.*, $x_{i:j} \in \{0,1\}$), one for each edge in the graph (*i.e.*, $M = |\mathcal{E}|$) where $x_{i:j} = 1$ means (i,j) is in the tour. We use $x_{i:j}$ and $x_{j:i}$ to refer to the same variable. Recall that for each node i, $\mathcal{E}(i,\cdot)$ denotes the edges adjacent to i. Define the factors of the factor-graph as follows

• **Local factors** represent the cost associated with each edge

$$\mathsf{f}_{i:j}(x_{i:j}) \;=\; \min\Big(1(x_{i:j} = 0), \mathbf{A}_{i,j}\Big) \quad \forall (i,j) \in \mathcal{E} \tag{5.1}$$

where $\mathsf{f}_{i:j}(x_{i:j})$ is either $\mathbf{A}_{i,j}$ or zero.

Any valid tour satisfies the following necessary and sufficient constraints – a.k.a. **Held-Karp constraints** [127]:

• **Degree factors** ensure that exactly two edges that are adjacent to each vertex are in the tour

$$\mathsf{f}_{\mathcal{E}(i,\cdot)}(\underline{x}_{\mathcal{E}(i,\cdot)}) \;=\; 1\Big(\big(\sum_{(i,j)\in\mathcal{E}(i,\cdot)} x_{i:j}\big) = 2\Big) \tag{5.2}$$

• **Subtour factors** ensure that there are no short-circuits – *i.e.*, there are no loops that contain strict subsets of nodes. To enforce this, for each $\mathcal{S} \subset \mathcal{V}$, define $\mathcal{E}(\mathcal{S},.) \stackrel{\text{def}}{=} \{(i,j) \in \mathcal{E} \mid i \in \mathcal{S}, j \notin \mathcal{S}\}$ to be the set of edges, with one end in $\mathcal{S}$ and the other end in $\mathcal{V} \setminus \mathcal{S}$. We need to have at least two edges leaving each subset $\mathcal{S}$. The following set of factors enforce these constraints

$$\mathsf{f}_{\mathcal{E}(\mathcal{S},\cdot)}(\underline{x}_{\mathcal{E}(\mathcal{S},\cdot)}) \;=\; 1\Big(\big(\sum_{(i,j)\in\mathcal{E}(\mathcal{S},\cdot)} x_{i:j}\big) \geq 2\Big) \quad \forall\, \mathcal{S} \subset \mathcal{V},\;\; \mathcal{S} \neq \emptyset \tag{5.3}$$

These three types of factors define a factor-graph, whose minimum energy configuration is the

smallest tour for TSP. Therefore we can use min-sum inference to obtain the optimal tour. Note that both subtour and degree constraints depend on large number of variables, however, due to sparsity they allow efficient linear time calculation of factor-to-variable messages; see section 5.2.1. The more significant computational challenge is that the complete TSP factor-graph has $O(2^N)$ subtour factors, one for each subset of variables. In section 5.2.1 we address this problem using factor-graph augmentation.

Simplified messages

In section 2.2.1, we introduced the K-of-N factors for min-sum inference. Both degree and subtour factors are different variations of this types of factor. For simplicity we work with normalized message $\widehat{\mathrm{p}}_{\mathrm{I}\to i:j} = \widehat{\mathrm{p}}_{\mathrm{I}\to i:j}(1) - \widehat{\mathrm{p}}_{\mathrm{I}\to i:j}(0)$, which is equivalent to assuming $\widehat{\mathrm{p}}_{\mathrm{I}\to i:j}(0) = 0\ \forall \mathrm{I}, i:j \in \partial \mathrm{I}$. The same notation is used for variable-to-factor message, and marginal belief. As before, we refer to the normalized marginal belief, $\widehat{\mathrm{p}}_{i:j} = \widehat{\mathrm{p}}(x_{i:j}=1) - \widehat{\mathrm{p}}(x_{i:j}=0)$ as bias.

Recall that a *degree constraint* for node i ($\mathrm{f}_{\mathcal{E}(i,\cdot)}$) depends on all the variables $x_{i:j}$ for edges (i,j) that are adjacent to i. Here we review the factor-to-variable message for min-sum BP

$$\widehat{\mathrm{p}}_{\mathcal{E}(i,\cdot)\to i:j}(x_{i:j}) \quad = \quad \min_{\underline{x}_{\setminus i:j}} \mathrm{f}_{\mathcal{E}(i,\cdot)}(\underline{x}_{\mathcal{E}(i,\cdot)}) \sum_{(i,k)\in\mathcal{E}(i,\cdot)\setminus(i,j)} \widehat{\mathrm{p}}_{i:k\to\mathcal{E}(i,\cdot)}(x_{i:k}) \tag{5.4}$$

We show that this message update simplifies to

$$\widehat{\mathrm{p}}_{\mathcal{E}(i,\cdot)\to i:j}(1) \quad = \quad \min\left(\widehat{\mathrm{p}}_{i:k\to\mathcal{E}(i,\cdot)} \mid (i,k)\in\mathcal{E}(i,\cdot)\setminus(i,j)\right) \quad \forall i\in\mathcal{V} \tag{5.5}$$

$$\widehat{\mathrm{p}}_{\mathcal{E}(i,\cdot)\to i:j}(0) \quad = \quad \min\left(\widehat{\mathrm{p}}_{i:k\to\mathcal{E}(i,\cdot)} + \widehat{\mathrm{p}}_{i:l\to\mathcal{E}(i,\cdot)} \mid (i,k),(i,l)\in\mathcal{E}(i,\cdot)\setminus(i,j)\right) \tag{5.6}$$

where for $x_{i:j} = 1$, in order to satisfy degree constraint $\mathrm{f}_{\mathcal{E}(i,\cdot)}(\underline{x}_{\mathcal{E}(i,\cdot)})$, only *one* other $x_{i:k}$ for $(i,k)\in\mathcal{E}(i,\cdot)\setminus(i,j)$ should be non-zero. On the other hand, we know that messages are normalized such that $\widehat{\mathrm{p}}_{i:j\to\mathcal{E}(i,\cdot)}(0) = 0$, which means they can be ignored in the summation of equation (5.4). For $x_{i:j} = 0$, in order to satisfy the constraint factor, *two* of the adjacent variables should have a non-zero value. Therefore we seek two such incoming messages with minimum values. Let $\min^k \mathcal{A}$ denote the k^{th} smallest value in the set $\mathcal{A}$ – *i.e.*, $\min\mathcal{A} \equiv \min^1\mathcal{A}$. We combine the updates above to get a "normalized message", $\widehat{\mathrm{p}}_{\mathcal{E}(i,\cdot)\to i:j}$, which is simply the negative of the second largest incoming message (excluding $\widehat{\mathrm{p}}_{\mathcal{E}(i,\cdot)\to i:j}$) to the degree factor $\mathrm{f}_{\mathcal{E}(i,\cdot)}$:

$$\widehat{\mathrm{p}}_{\mathcal{E}(i,\cdot)\to i:j} = \widehat{\mathrm{p}}_{\mathcal{E}(i,\cdot)\to i:j}(1) - \widehat{\mathrm{p}}_{\mathcal{E}(i,\cdot)\to i:j}(0) = -\min^2\{\widehat{\mathrm{p}}_{i:k\to\mathcal{E}(i,\cdot)} \mid (i,k)\in\mathcal{E}(i,\cdot)\setminus(i,j)\} \tag{5.7}$$

Following a similar procedure, factor-to-variable messages for *subtour factors* is given by

$$\widehat{\mathrm{p}}_{\mathcal{E}(\mathcal{S},\cdot)\to i:j} = -\max\left(0, \min^{2}\{\widehat{\mathrm{p}}_{i:k\to\mathcal{E}(\mathcal{S},\cdot)} \mid (i,k)\in\mathcal{E}(\mathcal{S},\cdot)\setminus(i,j)\}\right) \tag{5.8}$$

While we are searching for the minimum incoming message, if we encounter two messages with negative or zero values, we can safely assume $\widehat{\mathrm{p}}_{\mathcal{E}(\mathcal{S},\cdot)\to i:j} = 0$, and stop the search. This results in significant speedup in practice. Note that both equation (5.7) and equation (5.8) only need to calculate the second smallest incoming message to their corresponding factors, less the current outgoing message. In the asynchronous calculation of messages, this minimization should be repeated for each outgoing message. However in a factor-synchronous update, by finding *three* smallest incoming messages to each factor, we can calculate all the factor-to-variable messages at the same time.

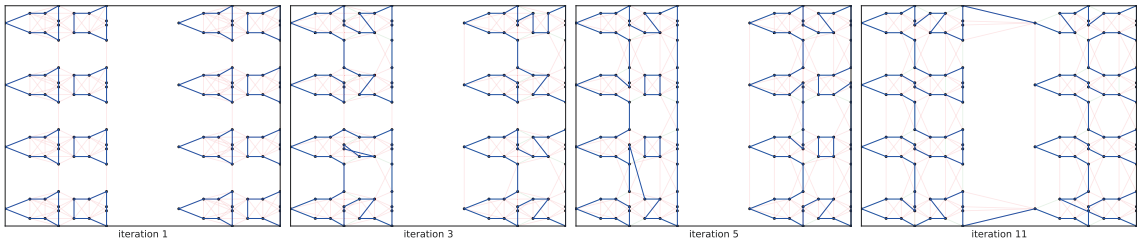


Figure 5.2: *The message passing results after each augmentation step for the complete graph of printing board instance from [260]. The blue lines in each figure show the selected edges at the end of message passing. The pale red lines show the edges with the bias that, although negative ($\widehat{\mathrm{p}}_{i:j} < 0$), were close to zero.*

Augmentation

To deal with the exponentially large number of subtour factors, we use the augmentation procedure of section 2.2.2. Starting with a factor-graph with no subtour factor, we find a solution $\underline{x}^*$ using min-sum BP. If the solution is feasible (has no subtours) we are done. Otherwise, we can find all subtours in $O(N)$ by finding connected components. We identify all the variables in each subtour as $\mathcal{S}\subset\mathcal{V}$ and add a subtour factor $\mathrm{f}_{\mathcal{E}(\mathcal{S},\cdot)}$ to ensure that this constraint is satisfied in the next iteration of augmentation. Here to speed up the message passing we reuse the messages from the previous augmentation step. Moreover in obtaining $\underline{x}^*$ from min-sum BP marginals $\widehat{\mathrm{p}}(x_i)$, we ensure that no degree constraint is violated (*i.e.*, each node has two neighbouring edges in $\underline{x}^*$). Figure 5.2 shows iterations of augmentation over a print board TSP instance from [260]. Algorithm 4 in the appendix gives details of our message passing solution to TSP, which also uses several other minor tricks to speed up the BP message updates.

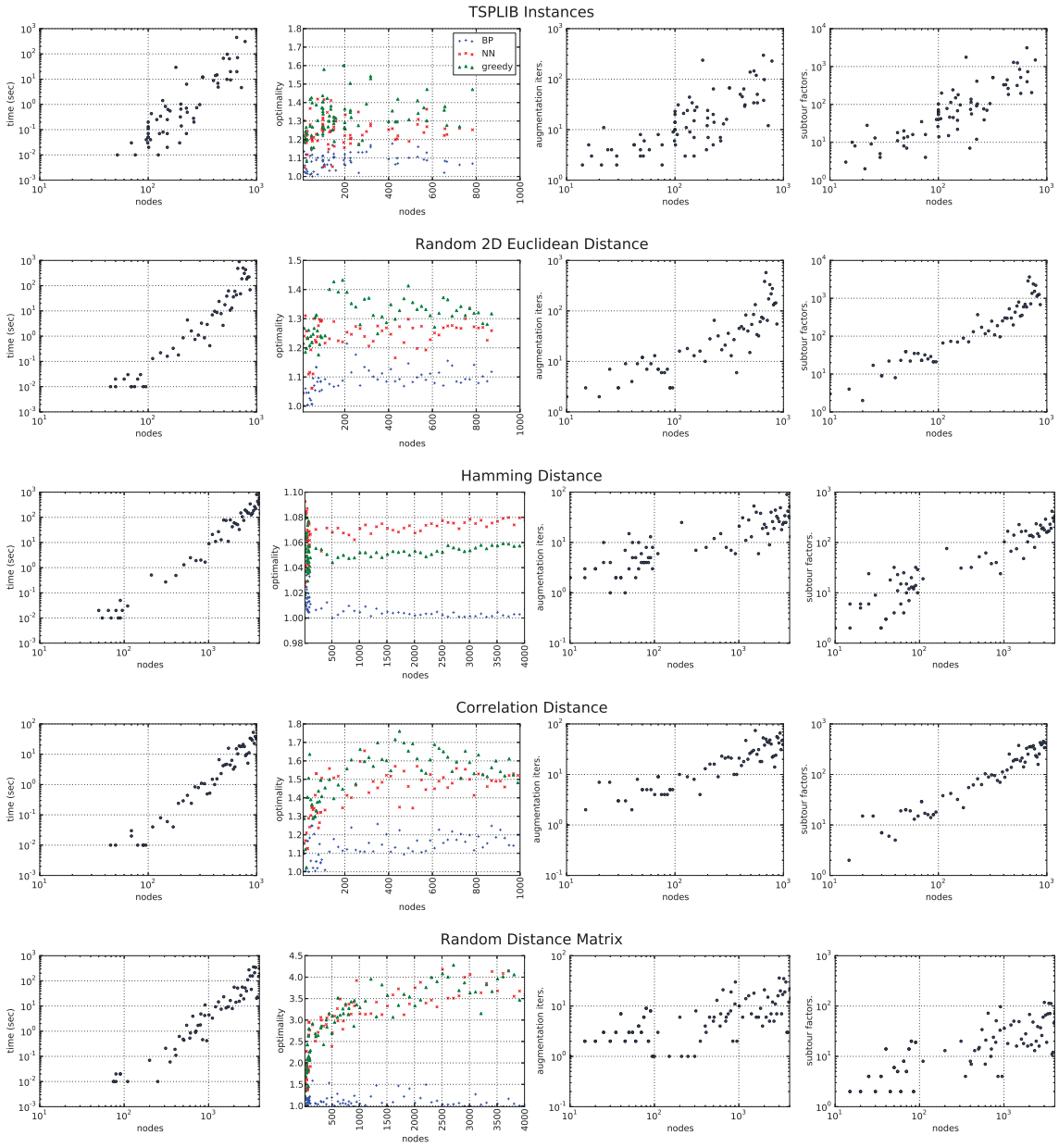


Figure 5.3: *Results of message passing for TSP on different benchmark problems. From left to right, the plots show: (a) running time, (b) optimality ratio (compared to Concorde), (c) iterations of augmentation and (d) number of subtours constraints – all as a function of number of nodes. The optimality is relative to the result reported by Concorde. Note that all plots except optimality are log-log plots where a linear trend shows a monomial relation ($y = ax^m$) between the values on the x and y axis, where the slope shows the power m.*

Experiments

Here we evaluate our method over five benchmark datasets[1]: **(I)** TSPLIB, which contains a variety of real-world benchmark instances, the majority of which are 2D or 3D Euclidean or geographic distances.[2] **(II)** Euclidean distance between random points in 2D. **(III)** Random (symmetric) distance matrices. **(IV)** Hamming distance between random binary vectors with fixed length (20 bits). This appears in applications such as data compression [153] and radiation hybrid mapping in genomics [29]. **(V)** Correlation distance between random vectors with 5 random features (*e.g.*, using TSP for gene co-clustering [69]). In producing random points and features as well as random distances (in (III)), we used uniform distribution over $[0, 1]$.

For each of these cases, we report the (a) run-time, (b) optimality, (c) number of iterations of augmentation and (d) number of subtour factors at the final iteration. In all of the experiments, we use Concorde [13] with its default settings to obtain the optimal solution.[3] The results in figure 5.3 (2nd column from left) reports the optimality ratio – *i.e.*, ratio of the tour found by message passing, to the optimal tour. To demonstrate the non-triviality of these instance, we also report the optimality ratio for two heuristics that have $(1 + \lceil \log_2(N) \rceil)/2$- optimality guarantees for metric instances [154]: (a) *nearest neighbour* heuristic ($O(N^2)$), which incrementally adds to any end of the current path the closest city that does not form a loop; (b) *greedy* algorithm ($O(N^2 \log(N))$), which incrementally adds a lowest cost edge to the current edge-set, while avoiding subtours.

All the plots in figure 5.3, except for the second column, are in log-log format. When using log-log plot, a linear trend shows a monomial relation between x and y axes – *i.e.*, $y = ax^m$. Here m indicates the slope of the line in the plot and the intercept corresponds to $\log(a)$. By studying the slope of the linear trend in the run-time (left column) in figure 5.3, we observe that, for almost all instances, message passing seems to grow with N^3 (*i.e.*, slope of ~ 3). Exceptions are TSPLIB instances, which seem to pose a greater challenge, and random distance matrices which seem to be easier for message passing. A similar trend is suggested by the number of subtour factors and iterations of augmentation, which has a slope of ~ 1, suggesting a linear dependence on N. Again the exceptions are TSPLIB instances that grow faster than N and random distance matrices that seem to grow sub-linearly.

Overall, we observe that augmentative message-passing is able to find near-optimal solutions in polynomial time. Although powerful branch-and-cut methods, such as Concorde, are able to

[1] In all experiments, we used the full graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, which means each iteration of message passing is $O(N^2\tau)$, where τ is the number of subtour factors. All experiments use $T_{\max} = 200$ iterations, $\epsilon_{\max} = \text{median}\{\mathbf{A}_{i,j} \mid i,j\}$ and damping with $\lambda = .2$. We used decimation, and fixed 10% of the remaining variables (out of N) per iteration of decimation. Note that here we are only fixing the top N variables with *positive* bias. The remaining $M - N$ variables are automatically clamped to zero. This increases the cost of message passing by an $O(\log(N))$ multiplicative factor, however it often produces better results.

[2] Geographic distance is the distance on the surface of the earth as a large sphere.

[3] For many larger instances, Concorde (with default setting and using CPLEX as LP solver) was not able to find the optimal solution. Nevertheless we used the upper bound on the optimal produced by Concord in evaluating our method.

exactly solve instances with several thousands of variables, their general run-time on random benchmark instances remains exponential [13, p495], while our approximation on random instances appears to be $O(N^3)$.

5.2.2 Using pairwise factors

Here we present an alternative factor-graph for finding permutations without subtours. This formulation has $O(N^2)$ factors and therefore the complete factor-graph remains tractable. However in practice, the min-sum inference over this factor-graph is not as effective as the augmentation approach. Therefore, here we use this factor-graph to solve min-max version of TSP, known as **bottleneck TSP** through sum-product reductions.[4]

Given an asymmetric distance matrix $\mathbf{A} \in \mathbb{R}^{N\times N}$, the task in the Bottleneck Traveling Salesman Problem (BTSP) is to find a tour of all N points such that the maximum distance between two consecutive cities in the tour is minimized [156]. Any constant-factor approximation for arbitrary instances of this problem is $\mathbb{NP}$-hard [242].

Let $\underline{x} = \{x_1, \dots, x_N\}$ denote the set of variables where $x_i \in \mathcal{X}_i = \{0, \dots, N-1\}$ represents the time-step at which node i is visited. We assume modular arithmetic (module N) on members of $\mathcal{X}_i$ – *e.g.*, $N \equiv 0 \mod N$ and $1 - 2 \equiv N - 1 \mod N$. For each pair x_i and x_j of variables, define the factor

$$\mathsf{f}_{\{i,j\}}(x_i, x_j) = \min\Big(1(|x_i - x_j| > 1), \max(\mathbf{A}_{i,j}, 1(x_i = x_j - 1)), \max(\mathbf{A}_{j,i}, 1(x_j = x_i - 1))\Big) \tag{5.9}$$

where the tabular form is

x_i \ x_j	0	1	$\cdots$	$N-2$	$N-1$		
0	∞	$\mathbf{A}_{i,j}$	$-\infty$	$\cdots$	$-\infty$	$-\infty$	$D_{j,i}$
1	$\mathbf{A}_{j,i}$	∞	$\mathbf{A}_{i,j}$	$\cdots$	$-\infty$	$-\infty$	$-\infty$
2	$-\infty$	$\mathbf{A}_{i,j}$	∞	$\cdots$	$-\infty$	$-\infty$	$-\infty$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\vdots$	$\vdots$
$N-3$	$-\infty$	$-\infty$	$-\infty$	$\cdots$	∞	$\mathbf{A}_{i,j}$	$-\infty$
$N-2$	$-\infty$	$-\infty$	$-\infty$	$\cdots$	$\mathbf{A}_{j,i}$	∞	$\mathbf{A}_{i,j}$
$N-1$	$\mathbf{A}_{i,j}$	$-\infty$	$-\infty$	$\cdots$	$-\infty$	$\mathbf{A}_{j,i}$	∞

Here, $\overset{\otimes}{1} = \infty$ on diagonal enteries ensures that $x_i \neq x_j$. Moreover $|x_i - x_j| > 1$ means cities i and j are not visited consecutively, so this factor has no effect ($\overset{\oplus}{1} = -\infty$). However if two cities are

[4] A binary-variable formulation with similar symantics is proposed in [306]. However the authors applied their message passing solution to an instance with only five cities.

visited one after the other, depending on whether i was visited before j or vicee-versa, $\mathbf{A}_{i,j}$ or $\mathbf{A}_{j,i}$ represent the distance between them. This factor can be easily converted to min-sum domains by replacing the identity values $-\infty, +\infty$ with $-\infty, 0$ in the tabular form above. Alternatively we can replace the identity, min and max operations in equation (5.9) with their corresponding min-sum functions.

Here we relate the min-max factor-graph above to a uniform distribution over Hamiltonian cycles and use its sum-product reduction to find solutions to Bottleneck TSP. Recall $\mathcal{G}(\mathbf{A}, y)$ is a graph in which there is a connection between node i and j iff $\mathbf{A}_{i,j} \leq y$.

Proposition 5.2.1. *For any distance matrix $\mathbf{A} \in \mathbb{R}^{N \times N}$, the p_y-reduction of the BTSP factor-graph above, defines a uniform distribution over the (directed) Hamiltonian cycles of $\mathcal{G}(\mathbf{A}, y)$.*

Proof. First note that p_y defines a uniform distribution over its support as its unnormalized value is only zero or one. Here w.l.o.g we distinguish between two Hamiltonian cycles that have a different starting point but otherwise represent the same tour. Consider the p_y-reduction of the pairwise factor of equation (5.9)

$$\mathrm{f}_{\{i,j\}}(x_i, x_j) = \mathbb{1}(|x_i - x_j| > 1) + \mathbb{1}(x_i = x_j - 1 \wedge \mathbf{A}_{i,j} \leq y) \tag{5.10}$$

$$+ \mathbb{1}(x_i = x_j + 1 \wedge \mathbf{A}_{j,i} \leq y) \tag{5.11}$$

• *Every Hamiltonian cycle over $\mathcal{G}(\mathbf{A}, y)$, defines a unique assignments $\underline{x}$ with $\mathrm{p}_y(\underline{x}) > 0$:* Given the Hamiltonian cycle $H = h_0, h_2, \ldots, h_{N-1}$ where $h_i \in \{1, \ldots, N\}$ is the i^{th} node in the path, for each i define $x_i = j \; s.t. \; h_j = i$. Now we show that all pairwise factors of equation (5.10) are non-zero for $\underline{x}$. Consider two variables x_i and x_j. If they are not consecutive in the Hamiltonian cycle then $\mathrm{f}_{\{i,j\}}(x_i, x_j) = \mathbb{1}(|x_i - x_j| > 1) > 0$. Now w.l.o.g. assume i and j are consecutive and x_i appears before x_j. This means $(i, j) \in \mathcal{E}$ and therefore $\mathbf{A}_{i,j} \leq y$, which in turn means $\mathrm{f}_{\{i,j\}}(x_i, x_j) = \mathbb{1}(x_i = x_j - 1 \wedge \mathbf{A}_{i,j} \leq y) > 0$ Since all pairwise factors are non-zero, $\mathrm{p}_y(\underline{x}) > 0$.
• *Every $\underline{x}$ for which $\mathrm{p}_y(x) > 0$, defines a unique Hamiltonian path over $\mathcal{G}(\mathbf{A}, y)$:* Given assignment $\underline{x}$, construct $H = h_0, \ldots, h_{N-1}$ where $h_i = j \; s.t. x_j = i$. Now we show that if $\mathrm{p}(\underline{x}) > 0$, H defines a Hamiltonian path. If $\mathrm{p}(\underline{x}) > 0$, for every two variables x_i and x_i, one of the indicator functions of equation (5.10) should evaluate to one. This means that first of all, $x_i \neq x_j$ for $i \neq j$, which implies H is well-defined and $h_i \neq h_j$ for $i \neq j$. Since all $x_i \in \{0, \ldots, N-1\}$ values are distinct, for each $x_i = s$ there are two variables $x_j = s-1$ and $x_k = s+1$ (recall that we are using modular arithmetic) for which the pairwise factor of equation (5.10) is non-zero. This means $\mathbf{A}_{j,i} \leq y$ and $\mathbf{A}_{i,k} \leq y$ and therefore $(j, i), (i, k) \in \mathcal{E}$ (the edge-set of $\mathcal{G}(\mathbf{A}, y)$). But by definition of H, $h_s = i$, $h_{s-1} = j$ and $h_{s+1} = k$ are consecutive nodes in H and therefore H is a Hamiltonian path. □

This proposition implies that we can use the sum-product reduction of this factor-graph to solve Hamiltonian cycle problems. The resulting factor-graph for Hamiltonian cycle problem is an

special case of our graph-on-graph technique, where the adjacency matrix of one graph is directly used to build factors in a second graph. Here, as the tabular form of the factor above suggests, the second graph is a simple loop of length N (see section 5.3).

As expected, due to sparse form of this pairwise factor, we can perform BP updates efficiently.

Claim 5.2.2. *The factor-to-variable BP messages for the sum-product reduction of factor of equation (5.9) can be obtained in $O(N)$.*

Proof. The p_y-reduction of the min-max factors of equation (5.9) is given by:

$$\mathrm{f}_{\{i,j\}}(x_i, x_j) = \mathbb{1}(\mathrm{f}_{\{i,j\}}(x_i, x_j) \leq y) \tag{5.12}$$

$$= \mathbb{1}(|x_i - x_j| > 1) + \mathbb{1}(x_i = x_j - 1 \wedge \mathbf{A}_{i,j} \leq y) \tag{5.13}$$

$$+ \mathbb{1}(x_i = x_j + 1 \wedge \mathbf{A}_{j,i} \leq y) \tag{5.14}$$

The matrix-form of this factor (depending on the order of $\mathbf{A}_{i,j}, \mathbf{A}_{j,i}, y$) takes several forms all of which are band-limited. Assuming the variable-to-factor messages are normalized (*i.e.*, $\sum_{x_i} \widehat{\mathrm{p}}_{j\to \mathrm{I}}(x_i) = 1$) the factor-to-variable message is given by

$$\widehat{\mathrm{p}}_{\{i,j\}\to i}(x_i) = 1 - \widehat{\mathrm{p}}_{j\to\{i,j\}}(x_i) +$$
$$\mathbb{1}(\mathbf{A}_{i,j} \leq y)(1 - \widehat{\mathrm{p}}_{j\to\{i,j\}}(x_i - 1)) +$$
$$\mathbb{1}(\mathbf{A}_{j,i} \leq y)(1 - \widehat{\mathrm{p}}_{j\to\{i,j\}}(x_i + 1))$$

Therefore the cost of calculating factor-to-variable message is that of normalizing the variable-to-factor message, which is $O(N)$. □

Since there are N^2 pairwise factors, this gives $O(N^3)$ time-complexity for each iteration of sum-product BP in solving the Hamiltonian cycle problem (*i.e.*, the sum-product reduction) and $O(N^3 \log(N))$ for bottleneck-TSP, where the $\log(N)$ factor is the cost of binary search (see section 1.4.2).

Figure 5.4 reports the average performance of message passing (over 10 instances) as well as a *lower bound* on the optimal min-max value for tours of different length (N). Here we report the results for random points in 2D Euclidean space as well as asymmetric random distance matrices. For Euclidean problems, the lower bound is the maximum over j of the distance of two closest neighbors to each node j. For the asymmetric random distance matrices, the maximum is over all the minimum length incoming edges and minimum length outgoing edges for each node.[5]

[5] If for one node the minimum length in-coming and out-going edges point to the same city, the second minimum length in-coming and out-going edges are also considered in calculating a tighter bound.

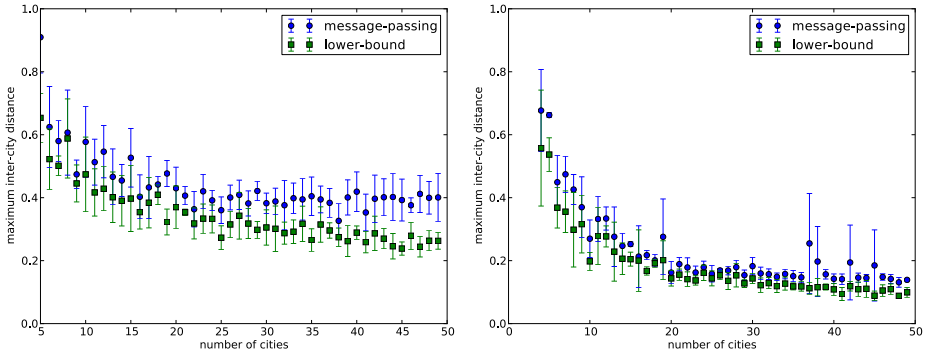


Figure 5.4: *The min-max solution (using sum-product reduction) for Bottleneck TSP with different number of cities (x-axis) for 2D Euclidean space (left) as well as asymmetric random distance matrices (right) with* $T = 5000$ *for Perturbed BP. The error-bars in all figures show one standard deviation over 10 random instances.*

5.3 Graph matching problems

Consider two graphs $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and $\mathcal{G}' = (\mathcal{V}', \mathcal{E}')$, with (weighted) adjacency matrices $\mathbf{A}$, $\mathbf{A}'$ respectively. Here we enumerate several of the most important problems over permutations based on two graphs [72]. Section 5.3.1 introduces the graphical model for the problems of (sub-graph) isomorphism and monomorphism. Then we study the message passing solution to graph homomorphism, use it to find symmetries in graphs. In section 5.3.5 we introduce a general factor-graph for graph alignment based on the previous work of Bradde et al. [44] and show how it can model other problems such as quadratic assignment problem and maximum common subgraph. The common idea in all these settings is that a "variation" of the adjacency matrix $\mathbf{A}'$ of graph $\mathcal{G}'$ is used as a pairwise factor over the edges (or/and non-edges) of graph $\mathcal{G}$ (with adjacency $\mathbf{A}$), defining a Markov network (a factor-graph with pairwise factors).

5.3.1 Sub-graph isomorphism

The graph **isomorphism** problem asks whether $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and $\mathcal{G}' = (\mathcal{V}', \mathcal{E}')$ are identical up to a permutation of nodes (written as $\mathcal{G} \cong \mathcal{G}'$). That is, it seeks a one-to-one mapping $\pi : \mathcal{V} \to \mathcal{V}'$ such that $(i,j) \in \mathcal{E} \Leftrightarrow (\pi(i), \pi(j)) \in \mathcal{E}'$. With some abuse of notation we also write $\pi(\mathcal{G}) = \mathcal{G}'$. Although, there have been polynomial time solutions to special instances of graph isomorphism problem [95], the general case has remained elusive. So much so that we do not know whether the problem is $\mathbb{NP}$-complete (*e.g.*, see [222]).

A permutation $\pi \in \mathcal{S}_N$ (recall $\mathcal{S}_N$ is the symmetric group) such that $\pi(\mathcal{G}) \cong \mathcal{G}$ is called an automorphism. The automorphisms of $\mathcal{G}$, under composition form a group, called the **automorphism group** $\mathrm{Aut}(\mathcal{G})$. The automorphism group also defines a natural notion of symmetry on the nodes of graphs. Here, the **orbit** of each node, is the set of nodes that are mapped to i in any automorphism – *i.e.*, $\mathrm{orbit}(i) \stackrel{\text{def}}{=} \{\pi(i) \mid \pi \in \mathrm{Aut}(\mathcal{G})\}$. The orbits partition the set of nodes $\mathcal{V}$ into

group of nodes that are in a sense symmetric, which makes them a prime candidate for defining symmetry in complex networks [198, 321].

Sub-graph isomorphism asks whether $\mathcal{G}'$ is isomorphic to a vertex induced subgraph of $\mathcal{G}$. – *i.e.*, it seeks an injective mapping $\pi : \mathcal{V}' \to \mathcal{V}$ where

$$(\pi(i), \pi(j)) \in \mathcal{E} \;\Leftrightarrow\; (i,j) \in \mathcal{E}' \quad \forall i,j \in \mathcal{V}'$$

When dealing with sub-graph morphisms, we assume that the mapping is from the smaller graph to the larger graph and therefore $|\mathcal{V}| \leq |\mathcal{V}'|$.

The factor-graph for subgraph isomorphism is defined as follows: We have one variable per $i \in \mathcal{V}$, where the domain of each variable x_i is $\mathcal{V}'$ – *i.e.*, $\underline{x} = \{x_i \in \mathcal{V}' \mid i \in \mathcal{V}\}$. The factor-graph has two types of pairwise factors:

- **Edge factors:** ensure that each edge in $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is mapped to an edge in $\mathcal{G}' = (\mathcal{V}', \mathcal{E}')$

$$\mathsf{f}_{\{i,j\}}(\underline{x}_{\{i,j\}}) = \mathbb{1}((x_i, x_j) \in \mathcal{E}') \quad \forall (i,j) \in \mathcal{E} \tag{5.15}$$

where assuming the tabular form of this factor for sum-product semiring is simply the adjacency matrix $\mathbf{A}' \in \{0,1\}^{|\mathcal{V}'| \times |\mathcal{V}'|}$ of $\mathcal{G}'$.

- **Non-edge factors:** ensure that each non-edge in $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ is mapped to a non-edge in $\mathcal{G}' = (\mathcal{V}', \mathcal{E}')$

$$\mathsf{f}_{\{i,j\}}(\underline{x}_{\{i,j\}}) = \mathbb{1}((x_i, x_j) \notin \mathcal{E}' \wedge x_i \neq x_j) \quad \forall i,j \in \mathcal{V}, i \neq j, (i,j) \notin \mathcal{E} \tag{5.16}$$

where again for sum-product semiring, the tabular form takes a simple form w.r.t. the binary valued adjacency matrix of $\mathcal{G}'$ – *i.e.*, $1 - \mathbf{A}'$. Using sum-product semiring, this fully connected Markov network defines a uniform distribution over the (subgraph) isomorphisms from $\mathcal{G}$ to $\mathcal{G}'$. We could use sum-product BP with decimation or perturbed BP to sample individual assignments $\underline{x}^* \sim \mathsf{p}(\underline{x})$. Here, each assignment $\underline{x} \equiv \pi$ is an (injective) mapping from $\mathcal{V}$ to $\mathcal{V}'$, where $x_i = j'$ means node $i \in \mathcal{V}$ is mapped to node $j' \in \mathcal{V}'$.

In particular, for $\mathcal{G} = \mathcal{G}'$, the integral is equal to the cardinality of the automorphism group $\mathsf{q}(\emptyset) = |\mathsf{Aut}(\mathcal{G})|$ and two nodes $i,j \in \mathcal{V}$ are in the same orbit iff the have the same marginals – *i.e.*,

$$\mathsf{p}(x_i) = \mathsf{p}(x_j) \;\Leftrightarrow\; i \in \mathsf{orbit}(j)$$

This also suggests a procedure for finding (approximate) symmetries in graphs. We can use sum-product BP to find marginals and group the nodes based on the similarity of their marginals. However, the cost of message passing in this graphical model is an important barrier in practice.

Claim 5.3.1. *Assuming* $|\mathcal{V}| \leq |\mathcal{E}| \leq |\mathcal{V}|^2$ *and* $|\mathcal{V}'| \leq |\mathcal{E}'| \leq |\mathcal{V}'|^2$ *and using variable syn-*

chronous update, the time complexity of each iteration of sum-product BP for subgraph isomorphism is $O(|\mathcal{V}|^2\ |\mathcal{E}'|)$.

Proof. First, we calculate the cost of sending sum-product BP messages through the edge and non-edge factors. For the edge factors ($\mathbf{A}'$), we can go through each row of the tabular form of the factor and multiply the non-zero entries with corresponding message – *i.e.*, $\mathrm{g}_{\{i,j\}}(x_i, x_j) = \widehat{\mathrm{p}}_{i\to\{i,j\}}(x_i)\mathrm{f}_{\{i,j\}}(x_i, x_j)$. Then we add the values in each column, to obtain the outgoing message – *i.e.*, $\widehat{\mathrm{p}}_{\{i,j\}\to j}(x_j) = \sum_{x_i} \mathrm{g}_{\{i,j\}}(x_i, x_j)$. This procedure depends on the number of non-zero entries – *i.e.*, $O(|\mathcal{E}'|)$. The procedure is similar for non-edge factors. Therefore the overall cost of sending sum-product BP messages through all $O(|\mathcal{V}|^2)$ factors is $O(|\mathcal{E}'|\ |\mathcal{V}|^2)$. Using variable synchronous update, calculating all variable-to-factor messages takes $O(|\mathcal{V}|^2|\mathcal{V}'|)$. Using the assumption of the claim, the overall cost is therefore $O(|\mathcal{E}'|\ |\mathcal{V}|^2)$. □

However it is possible to improve this complexity by considering sparse mappings. For example, we can restrict the domain of each variable x_i to all the nodes $j' \in \mathcal{V}'$ that have the same degree with node i, or furthermore to all the nodes that also have neighbours with same degree as the neighbours of node i.

5.3.2 Subgraph monomorphism and supermorphism

Sub-graph monomorphism relaxes the constraint of the subgraph isomorphism to

$$(i,j) \in \mathcal{E} \ \Rightarrow\ (\pi(i), \pi(j)) \in \mathcal{E}' \quad \forall i,j \in \mathcal{V} \tag{5.17}$$

where $\pi : \mathcal{V}' \to \mathcal{V}$ has to be injective – *i.e.*, nodes in $\mathcal{V}'$ are mapped to distinct nodes in $\mathcal{V}$. However, $\mathcal{G}'$ is allowed to cover a "subset" of edges in an induced subgraph of $\mathcal{G}$. We note here that *previous graphical models introduce in [43, 44] for isomorphism in fact define monomorphism.* The only difference between this factor-graph and that of sub-graph isomorphism is that the non-edge factors are replaced by the following uniqueness factors.

- **Uniqueness factors** are inverse Potts factors that ensure disconnected nodes are mapped to different nodes (– *i.e.*, the mapping is injective)

$$\mathrm{f}_{\{i,j\}}(\underline{x}_{\{i,j\}}) = \mathbb{1}(x_i \neq x_j) \quad \forall i.j \in \mathcal{V}, j \neq i, (i,j) \notin \mathcal{E} \tag{5.18}$$

Despite this difference, when $\mathcal{G} = \mathcal{G}'$ – that is when we are interested in automorphisms – or more generally when $|\mathcal{E}| = |\mathcal{E}'|$ and $|\mathcal{V}| = |\mathcal{V}'|$, the distribution defined by the monomorphism factor-graph is identical to that of isomorphism factor-graph.

Claim 5.3.2. *Assuming* $|\mathcal{V}| \leq |\mathcal{E}| \leq |\mathcal{V}|^2$ *and* $|\mathcal{V}'| \leq |\mathcal{E}'| \leq |\mathcal{V}'|^2$*, using variable synchronous update, the time complexity of each iteration of sum-product BP for sub-graph monomorphism* $O(|\mathcal{E}|\ |\mathcal{E}'| +$

$|\mathcal{V}'|\ |\mathcal{V}|^2$).[6]

Proof. The complexity of sending sum-product BP messages through the edge factors is $O(|\mathcal{E}'|)$ (see proof of claim 5.3.1). However, the uniqueness factors are inverse Potts factors and allow $O(|\mathcal{V}'|)$ calculation of messages. This means the overall cost of sending messages through all the factors is $O(|\mathcal{E}|\ |\mathcal{E}'|\ + |\mathcal{V}'|\ |\mathcal{V}|^2)$. The cost of calculating variable-to-factor messages using variable-synchronous update is also $O(|\mathcal{V}|^2|\mathcal{V}'|)$, which gives the overall complexity of $O(|\mathcal{E}|\ |\mathcal{E}'|\ + |\mathcal{V}'|\ |\mathcal{V}|^2)$ per BP iteration. □

So far we have seen two variations of mapping a graph $\mathcal{G}$ to a subgraph of $\mathcal{G}'$. In subgraph isomorphism, the image of $\mathcal{G}$ strictly agrees with a sub-graph of $\mathcal{G}'$. In monomorphism, the image is contained within a sub-graph. A third possibility is to ask for a mapping such that the image of $\mathcal{G}$ "contains" a sub-graph $\mathcal{G}'$:

$$(i,j) \in \mathcal{E} \ \Leftarrow\ (\pi(i),\pi(j)) \in \mathcal{E}' \quad \forall i,j \in \mathcal{V} \tag{5.19}$$

where again $\pi : \mathcal{V}' \to \mathcal{V}$ has to be injective. Due to lack of a better name, we call this mapping **subgraph supermorphism**.

The factor-graph for sub-graph supermorphism has two types of factors, both of which we have seen before: 1) The non-edge factors between non-edges (equation (5.16)) ensure that the image contains a sub-graph of $\mathcal{G}'$ while 2) uniqueness factors between the edges $(i,j) \in \mathcal{E}$ ensure that the mapping is injective.

Claim 5.3.3. *Assuming $|\mathcal{V}| \leq |\mathcal{E}| \leq |\mathcal{V}|^2$ and $|\mathcal{V}'| \leq |\mathcal{E}'| \leq |\mathcal{V}'|^2$, using variable synchronous update, the time complexity of each iteration of sum-product BP for subgraph supermorphism is $O((|\mathcal{V}^2| - |\mathcal{E}|)\ |\mathcal{E}'|\ + |\mathcal{V}'|\ |\mathcal{V}|^2)$*

Proof. The complexity of sending sum-product BP messages through the non-edge factors is $O(|\mathcal{E}'|)$ (see proof of claim 5.3.1) and each uniqueness factor require $O(|\mathcal{V}'|)$ computation. This means the overall cost of sending messages through all the factors is $O((|\mathcal{V}^2| - |\mathcal{E}|)\ |\mathcal{E}'|\ + |\mathcal{V}'|\ |\mathcal{E}|)$. The cost of calculating variable-to-factor messages using variable-synchronous update is also $O(|\mathcal{V}|^2|\mathcal{V}'|)$, which gives the overall complexity of $O((|\mathcal{V}^2| - |\mathcal{E}|)\ |\mathcal{E}'|\ + |\mathcal{V}'|\ |\mathcal{V}|^2)$ per BP iteration. □

5.3.3 Graph homomorphism

Graph homomorphism [129] further relaxes the constraint of the sub-graph monomorphism such that a homomorphic mapping can map two distinct nodes in $\mathcal{V}$ to the same node in $\mathcal{V}'$. However, equation (5.17) should hold, and therefore if two nodes in $\mathcal{V}'$ are adjacent, they should still be

[6] Bradde et al. [44] suggest a trick to reduce this time-complexity to $O((|\mathcal{E}|\ |\mathcal{V}|)|\mathcal{V}'|)$, but unfortunately details are omitted and we are unable to follow their route.

mapped to distinct and adjacent nodes in $\mathcal{V}$. Compared to other areas of graph theory, the study of rich structure and surprising properties of homomorphisms is relatively recent. Study of graph homomorphisms covers diverse areas including property testing [9], graph sequences and limits [37, 39, 40, 197] and constraint satisfaction problems [223]. Many of these areas are interested in "counting" the number of homomorphisms [38] from a graph $\mathcal{G}$ to $\mathcal{G}'$. As we see shortly, graph homomorphism reduces to many interesting CSPs and therefore it is not hard to show that it is $\mathbb{NP}$-complete. Moreover, counting the number of homomorphisms is $\#\mathbb{P}$-complete [86]. The set of all homomorphisms π of a graph $\mathcal{G}$ to itself –*i.e.*, its endomorphisms – under composition form **endomorphism monoid** (see definition 1.1.1 for monoid). Here, the identity element simply maps each element to itself. Our interest in endomorphism is because through the conjecture 5.3.4, we can use it (instead of automorphism) to approximate symmetries in graphs.

The graphical model representation of homomorphism has been investigated in different contexts [50, 52], but to our knowledge, message passing has not been previously used for counting and finding homomorphisms. The Markov network for homomorphism resembles that of isomorphism and monomorphism: The variables are $\underline{x} = \{x_i \mid i \in \mathcal{V}\}$ where $x_i \in \mathcal{V}'$, and the factor-graph *only contains edge-factors* of equation (5.15). Assuming $|\mathcal{V}'| \leq |\mathcal{E}'|$ and $|\mathcal{V}| \leq |\mathcal{E}|$, it is easy to see that the complexity of variable-synchronous message passing is $O(|\mathcal{E}|\ |\mathcal{E}'|)$, which makes this method very efficient for sparse graphs. This graphical model can be extended to represent homomorphism in weighted graphs [38]. For this define the edge-factors as

$$\mathrm{f}_{\{i,j\}}(\underline{x}_{\{i,j\}}) = \mathbf{A}_{i,j}^{\mathbf{A}'_{x_i,x_j}} \qquad \forall i,j \in \mathcal{V}, i \neq j, \mathbf{A}_{i,j} > 0$$

For small graphs we can exactly count the number of homomorphisms and endomorphisms. To evaluate the accuracy of message passing, we compared the BP estimates with the exact number of endomorphisms, for all isomorphically distinct graphs with $|\mathcal{V}| < 9$ (*i.e.*, $> 13,000$ instances); Figure 5.5 reports this result as well as the accuracy of BP marginals, suggesting that despite existence of short loops in these graphs BP estimates are relatively accurate.

Reduction to other CSPs

The relation between homomorphism and other CSPs is well-known [223]. Here, we review this relation between the factor-graphs of this section and other CSP factor-graphs we encountered in this thesis.

Coloring and clique cover: (see example 3.0.1 and section 3.4) K-coloring of $\mathcal{G}$ corresponds to finding a homomorphism from $\mathcal{G}$ to $\mathcal{K}_K$, the complete graph of order K. Similarly the homomorphism from the complement of $\mathcal{G}$ to $\mathcal{K}_k$ corresponds to K-clique-cover. The relation is also reflected in the corresponding factor-graphs as the adjacency matrix for $\mathcal{K}_K$ is the inverse Potts model, used in K-coloring.

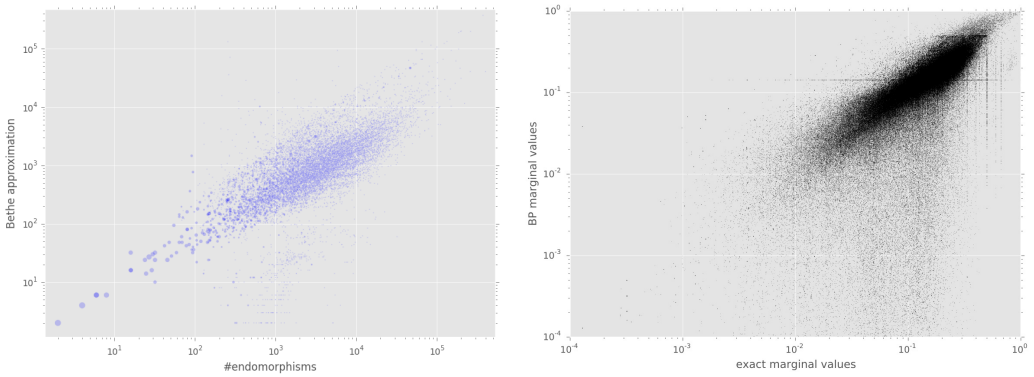


Figure 5.5: ***(left)*** *The number of endomorphisms for all distinct graphs up to 8 nodes compared to the BP integral. Here, larger disks represent graphs with smaller number of nodes.* ***(right)*** *Comparison of normalized BP marginals and the exact marginals for endomorphism graphical model.*

Clique problem and independent-set: (see section 3.6) Recall that K-clique problem corresponds to finding a clique of size K in $\mathcal{G}$. This is a special case of sub-graph isomorphism from $\mathcal{K}_K$ to $\mathcal{G}$ which is in this case identical to sub-graph monomorphism and homomorphism from $\mathcal{K}_K$ to $\mathcal{G}$. The (sum-product reduction of the) categorical-variable model of section 3.6.2 is a fully connected Markov network with edge factors that we used for isomorphism, monomorphism and homomorphism equation (5.15). Similarly, the K-independent set problem is equivalent to finding homomorphisms from $\mathcal{K}_K$ to the complement of $\mathcal{G}$. Independent set has an alternative relation with graph homomorphism: any homomorphism from $\mathcal{G}$ to a graph with two connected nodes and a self-loop on one of the nodes defines an independent set for $\mathcal{G}$.

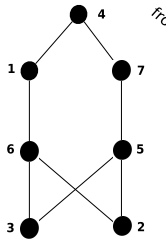
Hamiltonian cycle problem corresponds to sub-graph monomorphism from $C_{|\mathcal{V}|}$, the cycle of length $|\mathcal{V}|$, to $\mathcal{G}$. Alternatively, we can formulate it as subgraph supermorphism from $\mathcal{G}$ to $C_{|\mathcal{V}|}$. The sum-product reduction of our min-max formulation for bottleneck TSP in section 5.2.2 is indeed the factor-graph of sub-graph supermorphism from $\mathcal{G}$ to $C_{|\mathcal{V}|}$.

5.3.4 Finding symmetries

One may characterize a graph $\mathcal{G}$ using the "number" of homomorphism from/to other graphs $\mathcal{G}'$. This characterization is behind the application of graph homomorphism in property testing and definition of graph sequences. Let $\mathsf{Hom}(\mathcal{H},\mathcal{G})$ be the set of homomorphism from $\mathcal{H}$ to $\mathcal{G}$ – *i.e.*, the set of all assignments $\underline{x}$ where $\mathsf{p}(\underline{x}) > 0$. Let $\mathcal{H}_1,\ldots,\mathcal{H}_M$ be the sequence of all graphs whose number of nodes is at most $|\mathcal{V}|$. Then, the **Lovász vector** of $\mathcal{G}$ which is defined as

$$\underline{\mathsf{v}}(\mathcal{G}) \;=\; (|\mathsf{Hom}(\mathcal{H}_1,\mathcal{G})|,|\mathsf{Hom}(\mathcal{H}_2,\mathcal{G})|,\ldots,|\mathsf{Hom}(\mathcal{H}_M,\mathcal{G})|) \tag{5.20}$$

uniquely identifies $\mathcal{G}$ up to an isomorphism [196].



from \ to	1	2	3	4	5	6	7
1	26	0	0	26	0	0	26
2	20	4	4	18	6	6	20
3	20	4	4	18	6	6	20
4	26	0	0	26	0	0	26
5	25	0	0	16	6	6	25
6	25	0	0	16	6	6	25
7	26	0	0	26	0	0	26

Figure 5.6: *The table on the right shows the unnormalized endomorphism marginals for the graph on the left. Here row i corresponds to $\mathsf{q}(x_i)$ (normalization of which gives $\mathsf{p}(x_i)$), and the number at row i and column j is the number of times node i is mapped to j in an endomorphism. The total number of endomorphisms for this graph is $\mathsf{q}(\emptyset) = 78$. Here, the orbits are $\{2,3\},\{5,6\},\{1,7\},\{4\}$. However, $\mathsf{q}(x_1) = \mathsf{q}(x_7) = \mathsf{q}(x_4)$ – that is node 4 maps "to" other nodes with the same frequency as nodes 1 and 7. However, the mappings to node 4 (*i.e.*, the 4^{th} column of the table) remains different from the mappings to 1 and 7. Here, as predicted by conjecture 5.3.4, nodes with similar rows and columns belong to the same orbit.*

Here, rather than identifying a particular graph $\mathcal{G}$ within the set of all graphs, we are interested in identifying a node $i \in \mathcal{V}$ of a single graph $\mathcal{G}$ within the set of all nodes $\mathcal{V}$. Note that here both identifications are up to an isomorphism. Our objective is equivalent to finding the orbits of $\mathcal{G}$ and our approach in finding the orbits using graph homomorphism (rather than isomorphism) is founded on the following conjecture.

Conjecture 5.3.4. *Given the uniform distribution over the endomorphisms of $\mathcal{G}$:*

$$\mathsf{p}(\underline{x}) \propto \prod_{(i,j)\in\mathcal{E}} \mathbb{1}((x_i, x_j) \in \mathcal{E})$$

the necessary and sufficient condition for i and j to be in the same orbit is

$$\mathsf{p}(x_i) = \mathsf{p}(x_j) \;\; \textit{and} \;\; \forall k \in \mathcal{V} : \;\; \mathsf{p}(x_k = i) = \mathsf{p}(x_k = j) \quad \Leftrightarrow \quad \mathrm{orbit}(i) = \mathrm{orbit}(j)$$

Note that $\mathsf{p}(x_i = k)$ is simply the relative frequency of mapping of node i to node k in an endomorphism. Therefore this conjecture simply states that for node i and j to be equivalent up to an automorphism of $\mathcal{G}$, it is necessary and sufficient for them to have the same frequency of mapping to/from all other nodes of $\mathcal{G}$. While it is trivial to prove necessity ($\Leftarrow$), we found it difficult to prove the statement in the other direction.

Therefore, similar to other related conjectures [163, 204], we turn to **computational verification**. For this, we experimented with distinct graphs with up to 9 nodes (*i.e.*, $> 286,000$ instances). Here, we obtained the exact marginals $\mathsf{p}(x_i)$ $\forall i \in \mathcal{V}$ and constructed the orbits as suggested by conjecture 5.3.4. We then obtained the exact orbits using the software of McKay and Piperno [205]. In all cases two partitioning of the nodes to orbits were identical.

We also note that in conjecture 5.3.4 restricting the condition to $\mathsf{p}(x_i) = \mathsf{p}(x_j)$ – *i.e.*, similar frequency of mappings "to" other nodes – is not sufficient. Figure 5.6 shows a counter-example

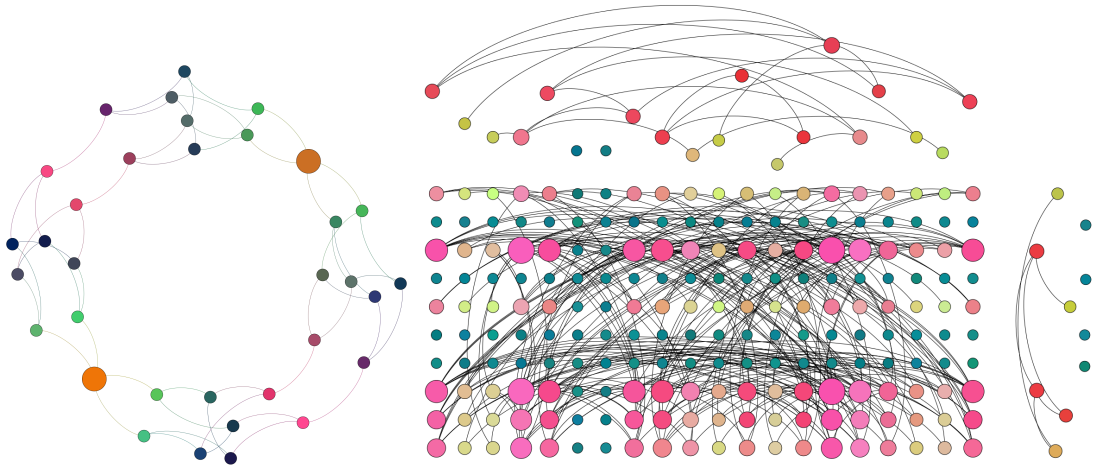


Figure 5.7: *Coloring of nodes by reducing the dimensionality of marginals to three dimensions of RGB using PCA with whitening.* ***(left)*** *The marginals are calculated using sum-product BP.* ***(right)*** *The marginals of the Kronecker graph is obtained using Gibbs sampling and annealing. The graph in the middle is the product of two graphs on the top and right – i.e., $\mathcal{G} = \mathcal{G}_1 \times \mathcal{G}_2$. Two nodes in $\mathcal{G}$ are connected iff the corresponding nodes in $\mathcal{G}_1$ and $\mathcal{G}_2$ are connected. Note that the product of each two colors produces the same color in the product graph and similarly colored nodes have similar neighbours.*

for this weaker condition.

Example 5.3.1. In figure 5.7(left) we used the homomorphism marginals to find approximate symmetries in a graph with visible symmetries, known as Thomassen graph. Here, after obtaining the marginals we used Principle Component Analysis (PCA) with whitening [226] to extract three values for RGB colors. As the figure suggests, this approach is able to identify similar nodes with similar colors.

For dense graphs, message passing is no longer accurate. In figure 5.7(right) we use Gibbs sampling with annealing to estimate the endomorphism marginals in **Kronecker product**[190] of two random graphs. Here again, we use PCA to color the nodes. Note that the algorithm is unaware of the product format. The choice of the product graph is to easily observe the fact that the product of similarly colored nodes produce similarly colored nodes in the product graph.

An alternative is to use spectral clustering on the matrix of marginals to obtain clusters. In a related context Krzakala et al. [180] use the matrix of non-backtracking random-walks to find symmetric clusters in stochastic block models. Note that the basic difference with our approach is the matrix used with the spectral clustering. Other notable matrices that are used within this context are the Laplacian matrix [299], the modularity matrix [228] and the Bethe hessian [272]. The relation between the clustering (in its conventional sense) and orbits of a graph is better understood in the extreme case: when clusters form isolated cliques, they are identical to orbits.

5.3.5 Graph alignment

The graph alignment problem can be seen as the optimization counterpart of the decision problems of isomprphism, monomorphism and homomorphism. In the past, different methods have tried to optimize a variety of different objectives [53, 72]. In the context of message passing two distinct approaches have been used: 1) Bayati et al. [26] propose a factor-graph for "sparse" graph alignment and show that it scales to very large instances. Here the term sparsity both refers to the number of edges of $\mathcal{G}$ and $\mathcal{G}'$ and also to the restricted possibility of matching nodes of $\mathcal{V}$ to $\mathcal{V}'$ – *i.e.*, each node in $\mathcal{V}$ can match only a few predetermined nodes in $\mathcal{V}'$. The factor-graph used by the authors resembles the binary version of the maximum bipartite matching factor-graph of section 5.1.1. 2) Bradde et al. [44] used the min-sum BP to minimize the number of misalignment in a factor-graph similar to that of graph monomorphism above. Here we follow their route, with the distinction that we suggest using the min-sum semiring with "sub-graph isomorphism" factor-graph and account for different matching costs using several tricks.

Here we consider a general objective function that evaluates the mapping $\pi : \mathcal{V}' \to \mathcal{V} \cup \{\text{NULL}\}$. We then show how to optimize this objective using max-sum inference in a factor-graph.

- Node matching preference for matching node $i \in \mathcal{V}$ to $j' \in \mathcal{V}'$: $\varphi(i,j') : \mathcal{V} \times \mathcal{V}' \to \mathbb{R}$.
- Edge matching preference for matching $(i,j) \in \mathcal{E}$ to $(i',j') \in \mathcal{E}'$: $\varsigma((i,j),(k',l')) : \mathcal{E} \times \mathcal{E}' \to \mathbb{R}$.
- Node merging preference for mapping nodes $i,j \in \mathcal{V}$ to the same node $k \in \mathcal{V}'$: $\vartheta(i,j,k') : \mathcal{V} \times \mathcal{V} \times \mathcal{V}' \to \mathbb{R}$.
- Node deletion preference $\delta(i) : \mathcal{V} \to \mathbb{R}$, is the penalty for ignoring the node $i \in \mathcal{V}$ – *i.e.*, mapping it to the NULL node.
- Edge deletion preference is the preference for dropping $(i,j) \in \mathcal{E}$: $\varpi(i,j) : \mathcal{E} \to \mathbb{R}$.
- Edge insertion preference is the preference for adding an edge $(i',j') \in \mathcal{E}'$, when it is not matched against any edge in $\mathcal{E}$: $\upsilon(i',j') : \mathcal{E}' \to \mathbb{R}$.

We can define these preferences in such a way that the optimal solution is also a solution to an interesting decision problem.

Example 5.3.2. The optimal alignments with the following parameters reproduce $\mathsf{Hom}(\mathcal{G},\mathcal{G}')$:

- node matching $\varphi(i,j') = 0 \quad \forall\, i \in \mathcal{V}, j' \in \mathcal{V}'$
- edge matching $\varsigma((i,j),(k',l')) = 1 \quad \forall\, (i,j) \in \mathcal{E}, (k',l') \in \mathcal{E}'$
- node merging $\vartheta(i,j,k') = 0 \quad \forall\, i,j \in \mathcal{V}, k' \in \mathcal{V}'$
- node deletion $\delta(i) = -\infty \quad \forall\, i \in \mathcal{V}$
- edge deletion $\varpi(i,j) = -\infty \quad \forall\, (i,j) \in \mathcal{E}$
- edge insertion $\upsilon(k',l') = 0 \quad \forall\, (k',l') \in \mathcal{E}'$

Alternatively, using positive and uniform node and edge matching preferences, if we set the merge cost to $-\infty$ and allow node deletion at zero cost, the optimal solution will be the **maximum common subgraph**. In particular, for maximum-edge common subgraph we use

- node matching $\varphi(i,j') = 0 \quad \forall\, i \in \mathcal{V}, j' \in \mathcal{V}'$
- edge matching $\varsigma((i,j),(k',l')) = 1 \quad \forall\, (i,j) \in \mathcal{E}, (k',l') \in \mathcal{E}'$
- node merging $\vartheta(i,j,k') = -\infty \quad \forall\, i,j \in \mathcal{V}, k' \in \mathcal{V}'$
- node deletion $\delta(i) = 0 \quad \forall\, i \in \mathcal{V}$
- edge deletion $\varpi(i,j) = 0 \quad \forall\, (i,j) \in \mathcal{E}$
- edge insertion $\upsilon(k',l') = 0 \quad \forall\, (k',l') \in \mathcal{E}'$

Given two weighted adjacency matrices $\mathbf{A}$ and $\mathbf{A}'$ (for $\mathcal{G}$ and $\mathcal{G}'$ respectively), where $\mathbf{A}_{i,j}$ is the "flow" between the facilities i and j, while $\mathbf{A}'_{k',l'}$ is the "distance" between the locations k' and l', the **quadratic assignment problem**, seeks a one-to-one mapping $\pi^* : \mathcal{V} \to \mathcal{V}'$ of facilities to locations in order to optimize the flow

$$\pi^* = \arg_\pi \max \sum_{i,j \in \mathcal{V}} \mathbf{A}_{i,j} \mathbf{A}'_{\pi(i),\pi(j)}$$

Here w.l.o.g. we assume all weights are positive and set the alignment preferences so as to optimize the quadratic assignment problem:

- node matching $\varphi(i,j') = 0 \quad \forall\, i \in \mathcal{V}, j' \in \mathcal{V}'$
- edge matching $\varsigma((i,j),(k',l')) = \mathbf{A}_{i,j}\mathbf{A}'_{k',l'} \quad \forall\, (i,j) \in \mathcal{E}, (k',l') \in \mathcal{E}'$
- node merging $\vartheta(i,j,k') = -\infty \quad \forall\, i,j \in \mathcal{V}, k' \in \mathcal{V}'$
- node deletion $\delta(i) = -\infty \quad \forall\, i \in \mathcal{V}$
- edge deletion $\varpi(i,j) = -\infty \quad \forall\, (i,j) \in \mathcal{E}$
- edge insertion $\upsilon(k',l') = -\infty \quad \forall\, (k',l') \in \mathcal{E}'$

Now we define the factors based on various alignment preferences. The factor-graph has one variable per node $i \in \mathcal{V}$: $\underline{x} = \{x_i \mid i \in \mathcal{V}\}$, where $x_i \in \mathcal{V}' \cup \{\text{NULL}\}$. Here, $x_i = \text{NULL}$ corresponds to ignoring this node in the mapping. The alignment factor-graph has three type of factors:

- **Local factors:** take the node matching preferences and node deletion preference into account:

$$\mathrm{f}_i(x_i) = \begin{cases} \delta(i) & x_i = \text{NULL} \\ \varphi(i,x_i) & \text{otherwise} \end{cases} \qquad \forall i \in \mathcal{V}$$

- **Edge factors:** are defined for each edge $(i,j) \in \mathcal{E}$ and partly account for edge matching, node

merging and edge deletion:

$$\mathsf{f}_i(x_i,x_j) \;=\; \begin{cases} 0 & x_i = \text{NULL} \;\vee\; x_j = \text{NULL} \\ \vartheta(i,j,x_i) & x_i = x_j \\ \varsigma((i,j),(x_i,x_j)) & (x_i,x_j) \in \mathcal{E}' \\ \varpi(i,j) & \text{otherwise} \end{cases} \qquad \forall (i,j) \in \mathcal{E}$$

- **Non-edge factors:** are defined for non-existing edge $i,j \neq i \in \mathcal{V}, (i,j) \notin \mathcal{E}$ and partly account for node merging and edge insertion:

$$\mathsf{f}_i(x_i,x_j) \;=\; \begin{cases} 0 & x_i = \text{NULL} \;\vee\; x_j = \text{NULL} \\ \vartheta(i,j,x_i) & x_i = x_j \\ \upsilon(x_i,x_j) & (x_i,x_j) \in \mathcal{E}' \\ 0 & \text{otherwise} \end{cases} \qquad \forall i,j \neq i \in \mathcal{V}, (i,j) \notin \mathcal{E}$$

This factor-graph in its general form is fully connected. The cost of max-sum message-passing through each of these factors is $O(|\mathcal{V}'|\ \log(\mathcal{V}'|))$, which means each iteration of variable synchronous max-sum BP is $O(|\mathcal{V}|^2\ |\mathcal{V}'|\ \log(\mathcal{V}'|))$. However, if the matching candidates are limited (a.k.a. sparse alignment), and the graphs $\mathcal{G}$ and $\mathcal{G}'$ are sparse, this cost can be significantly reduced in practice.

Example 5.3.3. Figure 5.8 shows a matching of E-coli metabolic network against a distorted version, where 50% of edges were removed and the same number of random edges were added. Then we generated 10 matching candidate for each node of the original graph, including the correct match and 9 other randomly selected nodes. We used graph alignment with the following preferences to match the original graph against the distorted version[7]

- node matching $\varphi(i,j') = 0 \quad \forall\ i \in \mathcal{V}, j' \in \mathcal{V}'$
- edge matching $\varsigma((i,j),(k',l')) = 1 \quad \forall\ (i,j) \in \mathcal{E}, (k',l') \in \mathcal{E}'$
- node merging $\vartheta(i,j,k') = -\infty \quad \forall\ i,j \in \mathcal{V}, k' \in \mathcal{V}'$
- node deletion $\delta(i) = -\infty \quad \forall\ i \in \mathcal{V}$
- edge deletion $\varpi(i,j) = 0 \quad \forall\ (i,j) \in \mathcal{E}$
- edge insertion $\upsilon(k',l') = 0 \quad \forall\ (k',l') \in \mathcal{E}'$

We observed that message passing using our factor-graph was able to correctly match all the nodes in two graphs.

[7] We used $T = 100$ initial iterations with damping parameter $\lambda = .2$. After this, we used decimation and fixed $\rho = 1\%$ of variables after each $T = 50$ iterations. The total run-time was less than 5 minutes.

Figure 5.8: *Matching the E-coli metabolic network against a highly distorted version using message passing. Here* 50% *of* $|\mathcal{E}| = 4306$ *edges in the original network (left) are* ***removed*** *and the same number of random edges are* ***added****, to produce the distorted network (*i.e., $|\mathcal{E}'| = |\mathcal{E}|$*). Each node had 10 random candidates for matching (including the correct choice) and message passing was able to identify the correct matching with* 100% *accuracy.*

Conclusion

This thesis studied a general form of inference in graphical models with an emphasis on algebraic abstractions. We organized an important subset of these inference problems under an inference hierarchy and studied the settings under which distributive law allows efficient approximations in the form of message passing. We investigated different methods to improve this approximation in loopy graphs using 1) variational formulation and loop correction; 2) survey propagation; 3) hybrid techniques. We then studied graphical modelling of combinatorial optimization problems under different modes of inference.

As with any other inference and optimization framework, graphical modeling has its pros and cons. The cons of using graphical models for combinatorial optimization are twofold a) implementing message passing procedures, when compared to other standard techniques such as using Integer Programming solvers, is more complex and time consuming. This is further complicated by b) the fact that there is no standard guideline for designing a factor-graph representation, so as to minimize the computational complexity or increase the quality of message passing solution. Indeed we used many tricks to efficiently approximate the solution to our problems; example include simplification of BP messages through alternative normalization, augmentation, variable and factor-synchronous message update, introduction of auxiliary variables, using damping and decimation *etc.*

On the other hand, when dealing with large scale and difficult optimization problems, one has to resort to conceptual and computational decomposition, and graphical modelling and message passing techniques are the immediate candidates. Message passing is mass parallelizable, scalable and often finds high-quality solutions. By providing factor-graphs for a diverse set of combinatorial problems, this thesis also was an attempt to establish the universality of message passing. Of course some of these problems better lend themselves to graphical modelling than some others, resulting in better computational complexity and quality of results. Table 5.1 summarizes some important information about the message passing solutions to combinatorial problems that are proposed or reviewed in this thesis.

Table 5.1: *Summary of message-passing solutions to combinatorial problems. The time complexity is for one iteration of message passing. We report different costs for different update schedules for the same problem. See the text for references. Here, N is the number of nodes and M is the number of constraints/factors.*

Problem	Semiring ops.	Complexity	Schedule	relation to others
Belief Propagation	*	$O(\sum_i \lvert\mathcal{X}_i\rvert\lvert\partial i\rvert^2 + \sum_\mathrm{I} \lvert\mathcal{X}_\mathrm{I}\rvert\lvert\partial \mathrm{I}\rvert)$	async.	-
	not $(\min, \max)$	$O(\sum_i \lvert\mathcal{X}_i\rvert\lvert\partial i\rvert + \sum_\mathrm{I} \lvert\mathcal{X}_\mathrm{I}\rvert\lvert\partial \mathrm{I}\rvert)$	v-sync.	
	*	$O(\sum_i \lvert\mathcal{X}_i\rvert\lvert\partial i\rvert + \sum_\mathrm{I} \lvert\mathcal{X}_\mathrm{I}\rvert)$	f-sync.	
Perturbed BP	$(+,\times)$	$O(\sum_i \lvert\mathcal{X}_i\rvert\lvert\partial i\rvert^2 + \sum_\mathrm{I} \lvert\mathcal{X}_\mathrm{I}\rvert\lvert\partial \mathrm{I}\rvert)$	async.	reduces to Gibbs samp. & BP
		$O(\sum_i \lvert\mathcal{X}_i\rvert\lvert\partial i\rvert + \sum_\mathrm{I} \lvert\mathcal{X}_\mathrm{I}\rvert\lvert\partial \mathrm{I}\rvert)$	v-sync.	
Survey Propagation	*	$O(\sum_i 2^{\lvert\mathcal{X}_i\rvert}\lvert\partial i\rvert^2 + \sum_\mathrm{I} 2^{\lvert\mathcal{X}_\mathrm{I}\rvert}\lvert\partial \mathrm{I}\rvert)$	async.	reduces to BP
K-satisfiability	$(+,\times)$	$O(K^2M + \sum_i \lvert\partial i\rvert^2)$	async.	-
		$O(K^2M)$	v-sync.	
		$O(KM)$	(f,v)-sync.	
K-coloring	$(+,\times)$	$O(K\sum_i \lvert\mathcal{E}(i,\cdot)\rvert^2)$	async.	K-clique-cover
		$O(K\lvert\mathcal{E}\rvert)$	v-sync.	
K-clique-cover	$(+,\times)$	$O(K\sum_i(N - \lvert\mathcal{E}(i,\cdot)\rvert)^2)$	a-sync.	$\equiv$ to K-coloring on $\mathcal{G}^c$
	$(+,\times)$	$O(K(N^2 - \lvert\mathcal{E}\rvert))$	v-sync.	
K-dominating-set	$(+,\times)$	$O(KN^2 + \sum_{i\in\mathcal{V}} \lvert\mathcal{E}(i,\cdot)\rvert^2 + \lvert\mathcal{E}(\cdot,i)\rvert^2)$	async.	-
& K-set-cover		$O(KN + \lvert\mathcal{E}\rvert)$	f-sync.	
min set-cover	$(\min,+)$	$O(\lvert\mathcal{E}\rvert)$	(f,v)-sync.	similar to K-median
K-independent-set	$(+,\times)$	$O(N^3)$	async.	**binary** var. model
& K-clique		$O(KN^2)$	f-sync.	$\equiv$ K-independent-set on $\mathcal{G}^c$
		$O(KN + \lvert\mathcal{E}\rvert)$	(f,v)-sync.	
K-packing	$(\min, \max)$	$O(\log(N)(KN + \lvert\mathcal{E}\rvert))$	(f,v)-sync.	$(+,\times)$ reduction $\equiv$ K-independent-set
	$(\min, \max)$	$O(KN + \lvert\mathcal{E}\rvert)$	(f,v)-sync.	min-max BP
K-independent-set	$(+,\times)$	$O(K^3N^2)$	async.	**categorical** var. model
& K-clique		$O(K^2N^2)$	v-sync.	$\equiv$ K-independent-set on $\mathcal{G}^c$
K-packing	$(\min, \max)$	$O(K^2N^2\log(N))$	v-sync.	$(+,\times)$ reduction $\equiv$ K-independent-set
max independent set	$(\max,+)$	$O(\lvert\mathcal{E}\rvert)$	(f,v)-sync.	
& min vertex cover				$\equiv$ max independent-set
sphere-packing	$(+,\times)$	$O(K^2 2^{2n})$	v-sync.	
(Hamming)		$O(K^3n + K^2n^2y)$	async.	-
n: digits		$O(K^2n^2y)$	v-sync.	
y: min dist.		$O(K^2ny)$	(f,v)-sync.	
K-medians	$(\min,+)$	$O(\lvert\mathcal{E}\rvert)$	f-sync.	a.k.a. affinity propagation
facility location	$(\min,+)$	$O(\lvert\mathcal{E}\rvert)$	f-sync.	
d-depth min span. tree	$(\min,+)$	$O(d\lvert\mathcal{E}\rvert)$	v-sync.	
prize-coll. Steiner tree	$(\min,+)$	$O(d\lvert\mathcal{E}\rvert)$	v-sync.	
K-clustering	$(\min, \max)$	$O(KN^2\log(N))$	v-sync.	$(+,\times)$ reduction $\equiv$ K-clique-cover
K-center	$(\min, \max)$	$O(\log(N)(KN + \lvert\mathcal{E}\rvert))$	f-sync.	$(+,\times)$ reduction $\equiv$ K-set-cover
		$O(KN + \lvert\mathcal{E}\rvert)$	min-max BP	
Modularity max	$(\min,+)$	-		clique model, using augmentation
		$O(K_{\max}N^2)$		Potts model
max matching	$(\min,+)$	$O(N^2)$	v-sync	-
& cycle cover				
bottleneck assignment	$(\min, \max)$	-	-	-
max b-matching	$(\min,+)$	$O(bN^2)$	v-sync	-
TSP	$(\min,+)$	$O(N^2\tau)$ or $\sim N^3$	(f,v)-sync	-
bottleneck TSP	$(\min, \max)$	$O(N^3\log(N))$	async	$(+,\times)$ reduction $\equiv$ Hamiltonian cycle
subgraph isomorphism	$(+,\times)$	$O(\lvert\mathcal{V}\rvert^2\lvert\mathcal{E}'\rvert)$	v-sync.	$\mathcal{G} \to \mathcal{G}'$
subgraph monomorphism	$(+,\times)$	$O(\lvert\mathcal{V}\rvert^2\ \lvert\mathcal{V}'\rvert + \lvert\mathcal{E}\rvert\ \lvert\mathcal{E}'\rvert)$	v-sync.	$\mathcal{G} \to \mathcal{G}'$
subgraph supermorphism	$(+,\times)$	$O((\lvert\mathcal{V}^2\rvert - \lvert\mathcal{E}\rvert)\ \lvert\mathcal{E}'\rvert\ + \lvert\mathcal{V}'\rvert\ \lvert\mathcal{V}\rvert^2)$	v-sync.	$\mathcal{G} \to \mathcal{G}'$
homomorphism	$(+,\times)$	$O(\lvert\mathcal{E}\rvert\ \lvert\mathcal{E}'\rvert)$	v-sync.	$\mathcal{G} \to \mathcal{G}'$
graph alignment	$(\max,+)$	$O(\lvert\mathcal{V}\rvert^2\ \lvert\mathcal{V}'\rvert\log(\lvert\mathcal{V}'\rvert))$	v-sync.	$\mathcal{G} \to \mathcal{G}'$ with general costs
max common sub-graph				a variation of graph alignment
quadratic assignment				a variation of graph alignment

Future work

Due to breadth of the models and problems that we covered in this thesis, our investigation lacks the deserved depth in many cases. This is particularly pronounced in the experiments. Moreover, we encountered many new questions and possibilities while preparing this thesis. Here, we enumerate some of the topics that demand a more in depth investigation in the future work

- Many of the problems that we discussed also have efficient LP relaxations that some times come with approximation guarantees. A comprehensive experimental comparison of message passing and LP relaxations for these problems, both in terms of speed and accuracy is highly desirable.
- Our algebraic approach to inference suggests that all message passing procedures discussed here, including survey propagation, are also applicable to the domain of complex numbers. Extensions to this domain not only may allow new applications (*e.g.*, using Fourier coefficients as factors) but may also produce better solutions to many problems that we have studied here (*e.g.*, in solving CSPs).
- Our study of using graph homomorphism and its application to finding symmetries is a work in progress. In particular its relation to other methods such as stochastic block models and spectral techniques needs further investigation.
- Although some preliminary results on Ising model suggested that using sum-product reductions for min-max inference performs much better than direct min-max message passing, an extensive comparison of these two approaches to min-max inference is missing in our analysis.
- In section 3.7, we noted that several optimization counterparts to CSPs allow using binary-search in addition to a direct optimization approach. While using binary search is more expensive for these problems, we do not know which approach will performs better in practice.

References

[1] D. Achioptas and G. B. Sorkin. Optimal myopic algorithms for random 3-sat. In *Foundations of Computer Science, 2000. Proceedings. 41st Annual Symposium on*, pages 590–600. IEEE, 2000. 79

[2] D. Achlioptas and A. Coja-Oghlan. Algorithmic barriers from phase transitions. *arXiv:0803.2122*, Mar. 2008. 79, 80

[3] D. Achlioptas, A. Naor, and Y. Peres. Rigorous location of phase transitions in hard optimization problems. *Nature*, 435(7043):759–764, June 2005. 80

[4] Y. Ahn, J. Bagrow, and S. Lehmann. Link communities reveal multiscale complexity in networks. *Nature*, 2010. 101

[5] H. Aissi, C. Bazgan, and D. Vanderpooten. Min–max and min–max regret versions of combinatorial optimization problems: A survey. *European J Operational Research*, 197(2):427–438, 2009. 29

[6] S. Aji and R. McEliece. The Generalized distributive law and free energy minimization. In *Allerton Conf*, 2001. 2, 48

[7] S. M. Aji and R. J. McEliece. The generalized distributive law. *Information Theory, IEEE Transactions on*, 46(2):325–343, 2000. 1

[8] S. M. Aji, G. B. Horn, and R. J. McEliece. On the Convergence of iterative decoding on graphs with a single cycle. In *Proc 1998 IEEE Int Symp Information Theory*, 1998. 36

[9] N. Alon and A. Shapira. Homomorphisms in graph property testing. In *Topics in discrete mathematics*, pages 281–313. Springer, 2006. 129

[10] D. J. Amit. *Modeling brain function: The world of attractor neural networks*. Cambridge University Press, 1992. 1, 69

[11] C. Andrieu, N. de Freitas, A. Doucet, and M. Jordan. An introduction to MCMC for machine learning. *Machine Learning*, 2003. 66

[12] D. Applegate. *The traveling salesman problem: a computational study*. Princeton, 2006. 43, 116

[13] D. Applegate, R. Bixby, V. Chvatal, and W. Cook. Concorde TSP solver, 2006. 116, 121, 122

[14] S. Arora and B. Barak. *Computational complexity: a modern approach*. Cambridge University Press, 2009. 18

[15] S. Arora, P. Raghavan, and S. Rao. Approximation schemes for euclidean k-medians and related problems. In *Proceedings of the thirtieth annual ACM symposium on Theory of computing*, pages 106–113. ACM, 1998. 101

[16] I. Averbakh. On the complexity of a class of combinatorial optimization problems with uncertainty. *Mathematical Programming*, 90(2):263–272, 2001. 29

[17] F. Bacchus. *Representing and reasoning with probabilistic knowledge: a logical approach to probabilities*. MIT press, 1991. 2

[18] M. Bailly-Bechet, S. Bradde, et al. Clustering with shallow trees. *J Statistical Mechanics*, 2009. 103

[19] M. Bailly-Bechet, A. Braunstein, and R. Zecchina. A prize-collecting steiner tree approach for transduction network inference. In *Computational Methods in Systems Biology*, pages 83–95. Springer, 2009. 103

[20] M. Bailly-Bechet, C. Borgs, et al. Finding undetected protein associations in cell signaling by belief propagation. *PPNAS*, 2011. 103

[21] E. Balas and P. Toth. Branch and bound methods. *The traveling salesman problem*, 1985. 116

[22] M. Bayati, D. Shah, and M. Sharma. Maximum weight matching via max-product belief propagation. In *ISIT*, pages 1763–1767. IEEE, 2005. 36, 114, 115

[23] M. Bayati, C. Borgs, A. Braunstein, J. Chayes, A. Ramezanpour, and R. Zecchina. Statistical mechanics of steiner trees. *Physical review letters*, 101(3):037208, 2008. 103

[24] M. Bayati, A. Braunstein, and R. Zecchina. A rigorous analysis of the cavity equations for the minimum spanning tree. *Journal of Mathematical Physics*, 49(12):125206, 2008. 104

[25] M. Bayati, D. Shah, and M. Sharma. Max-product for maximum weight matching: Convergence, correctness, and lp duality. *Information Theory, IEEE Transactions on*, 54(3):1241–1251, 2008. 115

[26] M. Bayati, D. F. Gleich, A. Saberi, and Y. Wang. Message-passing algorithms for sparse network alignment. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, 7(1):3, 2013. 133

[27] M. J. Beal, F. Falciani, Z. Ghahramani, C. Rangel, and D. L. Wild. A Bayesian approach to reconstructing genetic regulatory networks with hidden factors. *Bioinformatics*, 21(3):349–356, 2005. 1

[28] R. Bellman. Dynamic programming treatment of the travelling salesman problem. *JACM*, 1962. 116

[29] A. Ben-Dor and B. Chor. On constructing radiation hybrid maps. *J Computational Biology*, 1997. 121

[30] H. A. Bethe. Statistical theory of superlattices. *Proceedings of the Royal Society A Mathematical Physical and Engineering Sciences*, 150(871):552–575, 1935. 45

[31] I. Biazzo, A. Braunstein, and R. Zecchina. Performance of a cavity-method-based algorithm for the prize-collecting steiner tree problem on graphs. *Physical Review E*, 86(2):026706, 2012. 103

[32] B. Bidyuk and R. Dechter. Cutset sampling for Bayesian networks. *Journal of Artificial Intelligence Research*, 28(3):1–48, 2007. 50

[33] S. Bistarelli, U. Montanari, F. Rossi, T. Schiex, G. Verfaillie, and H. Fargier. Semiring-based csps and valued csps: Frameworks, properties, and comparison. *Constraints*, 4(3):199–240, 1999. 8, 24

[34] V. Blondel, J. Guillaume, et al. Fast unfolding of communities in large networks. *J Statistical Mechanics*, 2008. 107, 112

[35] H. L. Bodlaender. Classes of graphs with bounded tree-width, 1986. 48

[36] L. Bordeaux and E. Monfroy. Beyond np: Arc-consistency for quantified constraints. In *Principles and Practice of Constraint Programming-CP 2002*, pages 371–386. Springer, 2002. 21

[37] C. Borgs, J. Chayes, L. Lovász, V. T. Sós, B. Szegedy, and K. Vesztergombi. Graph limits and parameter testing. In *Proceedings of the thirty-eighth annual ACM symposium on Theory of computing*, pages 261–270. ACM, 2006. 129

[38] C. Borgs, J. Chayes, L. Lovász, V. T. Sós, and K. Vesztergombi. Counting graph homomorphisms. In *Topics in discrete mathematics*, pages 315–371. Springer, 2006. 129

[39] C. Borgs, J. T. Chayes, L. Lovász, V. T. Sós, and K. Vesztergombi. Convergent sequences of dense graphs i: Subgraph frequencies, metric properties and testing. *Advances in Mathematics*, 219(6): 1801–1851, 2008. 129

[40] C. Borgs, J. T. Chayes, L. Lovász, V. T. Sós, and K. Vesztergombi. Convergent sequences of dense graphs ii. multiway cuts and statistical physics. *Annals of Mathematics*, 176(1):151–219, 2012. 129

[41] E. Boros and P. L. Hammer. Pseudo-boolean optimization. *Discrete applied mathematics*, 123(1):155–225, 2002. 26

[42] S. Boyd and L. Vandenberghe. *Convex Optimization*, volume 25. Cambridge University Press, 2010. 46

[43] S. Bradde. *Statistical physics of subgraph identification problem*. PhD thesis, SISSA, 2010. 127

[44] S. Bradde, A. Braunstein, H. Mahmoudi, F. Tria, M. Weigt, and R. Zecchina. Aligning graphs and finding substructures by a cavity approach. *EPL (Europhysics Letters)*, 89(3):37009, 2010. 5, 125, 127, 128, 133

[45] U. Brandes, D. Delling, et al. on clustering. *IEEE KDE*, 2008. 107, 109

[46] A. Braunstein and R. Zecchina. Survey propagation as local equilibrium equations. *arXiv:cond-mat/0312483*, Dec. 2003. J. Stat. Mech., P06007 (2004). 58

[47] A. Braunstein, M. Mezard, M. Weigt, and R. Zecchina. Constraint satisfaction by survey propagation. *Physics*, (2):8, 2002. 58

[48] A. Braunstein, M. Mezard, and R. Zecchina. Survey propagation: an algorithm for satisfiability. *Random Structures and Algorithms*, 27(2):19, 2002. 58, 66, 83, 84

[49] 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. 83

[50] G. R. Brightwell and P. Winkler. Graph homomorphisms and phase transitions. *Journal of combinatorial theory, series B*, 77(2):221–262, 1999. 129

[51] R. L. Brooks. On colouring the nodes of a network. In *Mathematical Proceedings of the Cambridge Philosophical Society*, volume 37, pages 194–197. Cambridge Univ Press, 1941. 98

[52] A. Bulatov and M. Grohe. The complexity of partition functions. *Theoretical Computer Science*, 348 (2):148–186, 2005. 129

[53] H. Bunke. On a relation between graph edit distance and maximum common subgraph. *Pattern Recognition Letters*, 18(8):689–694, 1997. 133

[54] M. Castellana and L. Zdeborová. Adversarial satisfiability problem. *Journal of Statistical Mechanics: Theory and Experiment*, 2011(03):P03023, 2011. 98

[55] T. Castellani and A. Cavagna. Spin-glass theory for pedestrians. *Journal of Statistical Mechanics: Theory and Experiment*, 2005(05):P05012, May 2005. 46

[56] V. Černỳ. Thermodynamical approach to the traveling salesman problem: An efficient simulation algorithm. *J Optimization theory and Applications*, 1985. 116

[57] V. Chandrasekaran and N. Srebro. Complexity of inference in graphical models. *UAI*, 2008. 48

[58] M. Charikar, S. Guha, É. Tardos, and D. B. Shmoys. A constant-factor approximation algorithm for the k-median problem. In *Proceedings of the thirty-first annual ACM symposium on Theory of computing*, pages 1–10. ACM, 1999. 101

[59] P. Cheeseman, B. Kanefsky, and W. M. Taylor. Where the really hard problems are. In *Proceedings of the 12th IJCAI - Volume 1*, IJCAI'91, pages 331–337, San Francisco, CA, USA, 1991. Morgan Kaufmann Publishers Inc. 79

[60] M. Chertkov. Loop series for discrete statistical models on graphs. *{...} of Statistical Mechanics: Theory and {...}*, 2006. 50

[61] M. Chertkov and V. Y. Chernyak. Loop calculus in statistical physics and information science. *Physical Review E - Statistical, Nonlinear and Soft Matter Physics*, 73(6 Pt 2):4, 2006. 50

[62] M. Chertkov and A. B. Yedidia. Approximating the permanent with fractional belief propagation. *J. Mach. Learn. Res.*, 14(1):2029–2066, Jan. 2013. 114

[63] H. L. Chieu and W. S. Lee. Relaxed survey propagation: A sum-product algorithm for max-sat. In *AAAI*, pages 247–252, 2008. 98

[64] H. L. Chieu, W. S. Lee, and Y. W. Teh. Cooled and relaxed survey propagation for mrfs. In *Advances in Neural Information Processing Systems*, pages 297–304, 2007. 98

[65] P. S. Churchland and T. J. Sejnowski. *The computational brain*. The MIT press, 1992. 1

[66] J. Chuzhoy, S. Guha, E. Halperin, S. Khanna, G. Kortsarz, R. Krauthgamer, and J. S. Naor. Asymmetric k-center is log* n-hard to approximate. *JACM*, 2005. 105

[67] A. Clauset. Finding local community structure in networks. *Physical Review E*, 2005. 107, 112

[68] P. Clifford. Markov random fields in statistics. *Disorder in physical systems*, pages 19–32, 1990. 8

[69] S. Climer and W. Zhang. Take a walk and cluster genes: A TSP-based approach to optimal rearrangement clustering. In *ICML*, 2004. 121

[70] S. Cocco, O. Dubois, J. Mandler, and R. Monasson. Rigorous decimation-based construction of ground pure states for spin glass models on random lattices. *arXiv:cond-mat/0206239*, June 2002. 79

[71] A. Coja-Oghlan. On belief propagation guided decimation for random k-sat. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '11, pages 957–966. SIAM, 2011. 81

[72] D. Conte, P. Foggia, C. Sansone, and M. Vento. Thirty years of graph matching in pattern recognition. *International journal of pattern recognition and artificial intelligence*, 18(03):265–298, 2004. 125, 133

[73] M. Cooke, J. R. Hershey, and S. J. Rennie. Monaural speech separation and recognition challenge. *Computer Speech & Language*, 24(1):1–15, 2010. 1

[74] G. Cooper. The computational complexity of probabilistic inference using Bayesian belief networks. *Artificial Intelligence*, 42, 1990. 2

[75] T. M. Cover and J. A. Thomas. *Elements of Information Theory*, volume 6 of *Wiley Series in Telecommunications*. Wiley, 1991. 96

[76] C. Crick and A. Pfeffer. Loopy belief propagation as a basis for communication in sensor networks. In *Proceedings of the Nineteenth conference on Uncertainty in Artificial Intelligence*, pages 159–166. Morgan Kaufmann Publishers Inc., 2002. 1

[77] G. Dantzig, R. Fulkerson, and S. Johnson. Solution of a large-scale traveling-salesman problem. *J Operations Research society of America*, 1954. 42, 116

[78] A. Darwiche. Recursive conditioning. *Artificial Intelligence*, 126(1):5–41, 2001. 49

[79] B. A. Davey and H. A. Priestley. *Introduction to lattices and order*. Cambridge university press, 2002. 9

[80] C. P. De Campos. New complexity results for map in Bayesian networks. In *IJCAI*, volume 11, pages 2100–2106, 2011. 47

[81] N. De Freitas, P. Hø jen Sø rensen, M. Jordan, and S. Russell. Variational MCMC. In *Proceedings of the*, pages 120–127. Morgan Kaufmann Publishers Inc., 2001. 68

[82] A. Decelle, F. Krzakala, C. Moore, and L. Zdeborová. Asymptotic analysis of the stochastic block model for modular networks and its algorithmic applications. *Physical Review E*, 84(6):066106, 2011. 101

[83] R. Dechter and D. Larkin. Hybrid processing of beliefs and constraints. In *Proceedings of the Seventeenth conference on Uncertainty in artificial intelligence*, pages 112–119. Morgan Kaufmann Publishers Inc., 2001. 8

[84] D. L. Donoho, A. Maleki, and A. Montanari. Message-passing algorithms for compressed sensing. *Proceedings of the National Academy of Sciences*, 106(45):18914–18919, 2009. 1

[85] M. Dorigo and L. Gambardella. Ant colony system: a cooperative learning approach to the traveling salesman problem. *IEEE Evolutionary Computation*, 1997. 116

[86] M. Dyer and C. Greenhill. The complexity of counting graph homomorphisms. *Random Structures and Algorithms*, 17(3-4):260–289, 2000. 129

[87] M. E. Dyer and A. M. Frieze. A simple heuristic for the p-centre problem. *Operations Research Letters*, 3(6):285–288, 1985. 29, 105, 107

[88] D. J. Earl and M. W. Deem. Parallel Tempering: Theory, Applications, and New Perspectives. *Physical Chemistry Chemical Physics*, 7(23):21, 2005. 68

[89] F. Eaton and Z. Ghahramani. Choosing a variable to clamp. In *International Conference on Artificial Intelligence and Statistics*, pages 145–152, 2009. 49

[90] F. Eaton and Z. Ghahramani. Model reductions for inference: Generality of pairwise, binary, and planar factor graphs. *Neural computation*, 25(5):1213–1260, 2013. 26

[91] B. Eckmann and P. J. Hilton. Group-like structures in general categories (I) multiplications and comultiplications. *Mathematische Annalen*, 145(3):227–255, 1962. 38

[92] J. Edmonds and D. Fulkerson. Bottleneck extrema. *J Combinatorial Theory*, 1970. 29

[93] S. F. Edwards and P. W. Anderson. Theory of spin glasses. *Journal of Physics F Metal Physics*, 5(5): 965–974, 1975. 13

[94] G. Elidan and I. McGraw. Residual belief propagation: Informed scheduling for asynchronous message passing. In *Proceedings of the Twenty- {...}*, 2006. 36

[95] D. Eppstein. Subgraph isomorphism in planar graphs and related problems. In *Proceedings of the sixth annual ACM-SIAM symposium on Discrete algorithms*, pages 632–640. Society for Industrial and Applied Mathematics, 1995. 125

[96] S. Ermon, C. P. Gomes, A. Sabharwal, and B. Selman. Optimization with parity constraints: From binary codes to discrete integration. *arXiv preprint arXiv:1309.6827*, 2013. 26

[97] S. Ermon, C. Gomes, A. Sabharwal, and B. Selman. Low-density parity constraints for hashing-based discrete integration. In *Proceedings of The 31st International Conference on Machine Learning*, pages 271–279, 2014. 26

[98] P. F. Felzenszwalb and D. P. Huttenlocher. Efficient belief propagation for early vision. *International journal of computer vision*, 70(1):41–54, 2006. 1

[99] M. E. Fisher. On the dimer solution of planar ising models. *Journal of Mathematical Physics*, 7(10): 1776–1781, 1966. 26

[100] G. D. Forney Jr. Codes on graphs: normal realizations. *Information Theory, IEEE Transactions on*, 47 (2):520–548, 2001. 8

[101] W. T. Freeman, E. C. Pasztor, and O. T. Carmichael. Learning low-level vision. *International journal of computer vision*, 40(1):25–47, 2000. 1

[102] B. Frey and D. Dueck. Clustering by passing messages between data points. *Science*, 2007. 101, 103

[103] Y. Fu and P. W. Anderson. Application of statistical mechanics to NP-complete problems in combinatorial optimisation. *Journal of Physics A: Mathematical and General*, 19(9):1605–1620, June 1986. 2, 79

[104] D. Fulkerson. Flow networks and combinatorial operations research. *The American Mathematical Monthly*, 73(2):115–138, 1966. 29

[105] M. H. Gail, J. H. Lubin, and L. V. Rubinstein. Likelihood calculations for matched case-control studies and survival studies with tied death times. *Biometrika*, 68(3):703–707, 1981. 40

[106] R. G. Gallager. Low-density parity-check codes. *Information Theory, IRE Transactions on*, 8(1):21–28, 1962. 1, 36

[107] M. R. Garey and D. S. Johnson. *Computers and Intractability: A Guide to the Theory of NP-Completeness*, volume 44 of *A Series of Books in the Mathematical Sciences*. W. H. Freeman, 1979. 76, 89

[108] A. E. Gelfand and A. F. M. Smith. Sampling-Based Approaches to Calculating Marginal Densities. *Journal of the American Statistical Association*, 85(410):398–409, 1990. 49

[109] S. Geman and D. Geman. Stochastic Relaxation, Gibbs Distributions, and the Bayesian Restoration of Images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, PAMI-6(6):721–741, 1984. 68

[110] S. Gershman, M. Hoffman, and D. Blei. Nonparametric variational inference. June 2012. 68

[111] I.-E. Givoni. *Beyond Affinity Propagation: Message Passing Algorithms for Clustering*. PhD thesis, University of Toronto, 2012. 103

[112] I. E. Givoni and B. J. Frey. A binary variable model for affinity propagation. *Neural computation*, 21 (6):1589–1600, 2009. 101

[113] A. Globerson and T. Jaakkola. Approximate inference using conditional entropy decompositions. In *International Conference on Artificial Intelligence and Statistics*, pages 130–138, 2007. 46

[114] A. Globerson and T. S. Jaakkola. Fixing max-product: Convergent message passing algorithms for map lp-relaxations. In *Advances in neural information processing systems*, pages 553–560, 2008. 46

[115] D. E. Goldberg and R. Lingle. Alleles, loci, and the traveling salesman problem. In *Genetic algorithms and Applications*, 1985. 116

[116] G. H. Golub and C. F. Van Loan. *Matrix computations*, volume 3. JHU Press, 2012. 113

[117] V. Gomez, J. M. Mooij, and H. J. Kappen. Truncating the loop series expansion for belief propagation. *Journal of Machine Learning Research*, 8:1987–2016, Sept. 2007. 50

[118] R. E. Gomory et al. Outline of an algorithm for integer solutions to linear programs. *Bulletin of the American Mathematical Society*, 64(5):275–278, 1958. 42, 116

[119] T. F. Gonzalez. Clustering to minimize the maximum intercluster distance. *Theoretical Computer Science*, 38:293–306, 1985. 29, 105

[120] J. Grefenstette, R. Gopal, B. Rosmaita, and D. Van Gucht. Genetic algorithms for the traveling salesman problem. In *Genetic algorithms and Applications*, 1985. 116

[121] O. Gross. The bottleneck assignment problem. Technical report, DTIC Document, 1959. 29, 114

[122] M. Grotschel and L. Lovász. Combinatorial optimization. *Handbook of combinatorics*, 2:1541–1597, 1995. 1

[123] R. Gupta, A. A. Diwan, and S. Sarawagi. Efficient inference with cardinality-based clique potentials. In *Proceedings of the 24th international conference on Machine learning*, pages 329–336. ACM, 2007. 39

[124] W. K. Hastings. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika*, 57(1):97–109, 1970. 67

[125] T. Hazan and T. Jaakkola. On the partition function and random maximum a-posteriori perturbations. *arXiv preprint arXiv:1206.6410*, 2012. 26

[126] T. Hazan and A. Shashua. Convergent Message-Passing Algorithms for Inference over General Graphs with Convex Free Energies. *Science*, 51(11):264–273, 2008. 46

[127] M. Held and R. Karp. The traveling-salesman problem and minimum spanning trees. *Operations Research*, 1970. 117

[128] M. Held and R. M. Karp. A dynamic programming approach to sequencing problems. *Journal of the Society for Industrial & Applied Mathematics*, 10(1):196âĂŞ210, 1962. 116

[129] P. Hell and J. Nesetril. *Graphs and homomorphisms*. Oxford University Press, 2004. 128

[130] K. Helsgaun. General k-opt submoves for the lin–kernighan tsp heuristic. *Mathematical Programming Computation*, 2009. 116

[131] J. R. Hershey, S. J. Rennie, P. A. Olsen, and T. T. Kristjansson. Super-human multi-talker speech recognition: A graphical modeling approach. *Computer Speech & Language*, 24(1):45–66, 2010. 1

[132] T. Heskes. Stable fixed points of loopy belief propagation are minima of the Bethe free energy. *Advances in Neural Information Processing Systems NIPS2002*, 14:343–350, 2003. 45

[133] T. Heskes. Convexity arguments for efficient minimization of the Bethe and Kikuchi free energies. *JAIR*, 26, 2006. 46, 56

[134] T. Heskes. Convexity Arguments for Efficient Minimization of the Bethe and Kikuchi Free Energies, 2006. 46

[135] D. S. Hochbaum and D. B. Shmoys. A best possible heuristic for the k-center problem. *Mathematics of operations research*, 10(2):180–184, 1985. 105

[136] D. S. Hochbaum and D. B. Shmoys. A unified approach to approximation algorithms for bottleneck problems. *Journal of the ACM (JACM)*, 33(3):533–550, 1986. 29

[137] R. A. Howard and J. E. Matheson. Influence diagrams. *Decision Analysis*, 2(3):127–143, 2005. 8

[138] B. Huang and T. Jebara. Loopy belief propagation for bipartite maximum weight b-matching. In *AI and Statistics*, 2007. 114

[139] B. Huang and T. Jebara. Approximating the permanent with belief propagation. *arXiv preprint arXiv:0908.1769*, 2009. 18, 114

[140] M. Ibrahimi, A. Javanmard, Y. Kanoria, and A. Montanari. Robust max-product belief propagation. In *Signals, Systems and Computers (ASILOMAR), 2011 Conference Record of the Forty Fifth Asilomar Conference on*, pages 43–49. IEEE, 2011. 29, 39

[141] A. Ihler. Particle belief propagation. *{...} Conference on Artificial Intelligence and Statistics*, 2009. 68

[142] A. Ihler and M. David. Nonparametric belief propagation. 2003. 68

[143] A. T. Ihler, J. W. Fisher, R. L. Moses, and A. S. Willsky. Nonparametric belief propagation for self-localization of sensor networks. *Selected Areas in Communications, IEEE Journal on*, 23(4):809–819, 2005. 1

[144] M. Isard. PAMPAS: real-valued graphical models for computer vision, 2003. 68

[145] S. Janson, T. Luczak, and V. F. Kolchin. Random graphs. *Bull. London Math. Soc*, 33:363âĂŞ383, 2001. 82

[146] E. T. Jaynes. *Probability theory: the logic of science*. Cambridge university press, 2003. 2

[147] C. S. Jensen, A. Kong, and U. Kjaerulff. Blocking-Gibbs Sampling in Very Large Probabilistic Expert Systems. *International Journal of Human-Computer Studies*, 42:647–666, 1995. 68

[148] F. V. Jensen, S. L. Lauritzen, and K. G. Olesen. Bayesian updating in causal probabilistic networks by local computations. *Computational Statistics Quarterly*, 4(4):269–282, 1990. 48

[149] M. Jerrum and A. Sinclair. The markov chain monte carlo method: an approach to approximate counting and integration. *Approximation algorithms for NP-hard problems*, pages 482–520, 1996. 114

[150] M. Jerrum, A. Sinclair, and E. Vigoda. A polynomial-time approximation algorithm for the permanent of a matrix with nonnegative entries. *Journal of the ACM (JACM)*, 51(4):671–697, 2004. 114

[151] M. R. Jerrum, L. G. Valiant, and V. V. Vazirani. Random generation of combinatorial structures from a uniform distribution. *Theoretical Computer Science*, 43:169–188, 1986. 25

[152] J. Jiang, P. Rai, and H. Daume. Message-passing for approximate map inference with latent variables. In *Advances in Neural Information Processing Systems*, pages 1197–1205, 2011. 47

[153] D. Johnson, S. Krishnan, J. Chhugani, S. Kumar, and S. Venkatasubramanian. Compressing large Boolean matrices using reordering techniques. In *VLDB*, 2004. 121

[154] D. S. Johnson and L. A. McGeoch. The traveling salesman problem: A case study in local optimization. *Local search in combinatorial optimization*, page 215âĂŞ310, 1997. 121

[155] V. Jojic, S. Gould, and D. Koller. Accelerated dual decomposition for map inference. In *Proceedings of the 27th International Conference on Machine Learning (ICML-10)*, pages 503–510, 2010. 46

[156] S. N. Kabadi and A. P. Punnen. The bottleneck tsp. In *The Traveling Salesman Problem and Its Variations*, pages 697–735. Springer, 2004. 122

[157] M. Kaminski, V. V. Lozin, and M. Milanic. Recent developments on graphs of bounded clique-width. *Discrete Applied Mathematics*, 157(12):2747–2761, 2009. 48

[158] V. Kann. *On the approximability of NP-complete optimization problems*. PhD thesis, Royal Institute of Technology Stockholm, 1992. 89

[159] R. M. Karp. *Reducibility among combinatorial problems*. Springer, 1972. 88

[160] H. A. Kautz, A. Sabharwal, and B. Selman. Incomplete algorithms., 2009. 32, 98

[161] M. Kearns. Graphical games. *Algorithmic game theory*, 3:159–180, 2007. 39

[162] M. Kearns, M. L. Littman, and S. Singh. Graphical models for game theory. In *Proceedings of the Seventeenth conference on UAI*, pages 253–260. Morgan Kaufmann Publishers Inc., 2001. 29

[163] P. J. Kelly et al. A congruence theorem for trees. *Pacific J. Math*, 7(0957):961–968, 1957. 131

[164] S. Khuller and Y. J. Sussmann. The capacitated k-center problem. *SIAM Journal on Discrete Mathematics*, 13(3):403–418, 2000. 29

[165] R. Kikuchi. A Theory of Cooperative Phenomena. *Physical Review*, 81(6):988–1003, 1951. 48, 49

[166] S. Kirkpatrick, M. Vecchi, et al. Optimization by simmulated annealing. *Science*, 1983. 116

[167] T. Kloks and H. L. Bodlaender. Only few graphs have bounded treewidth. *RUUCS*, 1992. 48

[168] P. Kohli and P. H. S. Torr. Graph Cuts for Minimizing Robust Higher Order Potentials. *Computing*, 24(7):996–1000, 2008. 39

[169] P. Kohli, M. P. Kumar, and P. H. S. Torr. P3 & Beyond: Solving Energies with Higher Order Cliques. *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2007. 39

[170] P. Kohli, L. Ladický, and P. H. S. Torr. Robust Higher Order Potentials for Enforcing Label Consistency. *International Journal of Computer Vision*, 82(3):302–324, Jan. 2009. 39

[171] D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*, volume 2009 of *Adaptive Computation and Machine Learning*. The MIT Press, 2009. 48

[172] D. Koller, U. Lerner, and D. Angelov. A general algorithm for approximate inference and its application to hybrid bayes nets. pages 324–333, July 1999. 68

[173] V. Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 28(10):1568–1583, 2006. 36

[174] N. Komodakis. Beyond pairwise energies: Efficient optimization for higher-order MRFs. *computer.org*, 2009. 39

[175] N. Komodakis and N. Paragios. Mrf energy minimization and beyond via dual decomposition. *IEEE Transactions on Pattern {...}*, 2010. 46

[176] N. Komodakis, N. Paragios, and G. Tziritas. Mrf optimization via dual decomposition: Message-passing revisited. In *Computer Vision, 2007. ICCV 2007. IEEE 11th International Conference on*, pages 1–8. IEEE, 2007. 46

[177] L. Kroc, A. Sabharwal, and B. Selman. Survey Propagation Revisited. *23rd UAI*, pages 217–226, 2002. 58

[178] A. Krogh, B. Larsson, G. Von Heijne, and E. L. Sonnhammer. Predicting transmembrane protein topology with a hidden markov model: application to complete genomes. *Journal of molecular biology*, 305(3):567–580, 2001. 1

[179] F. Krzakala, A. Montanari, F. Ricci-Tersenghi, G. Semerjian, and L. Zdeborova. Gibbs states and the set of solutions of random constraint satisfaction problems. *Proceedings of the National Academy of Sciences of the United States of America*, 104(25):10318âĂŞ10323, 2007. 79, 80, 82

[180] F. Krzakala, C. Moore, E. Mossel, J. Neeman, A. Sly, L. Zdeborová, and P. Zhang. Spectral redemption in clustering sparse networks. *Proceedings of the National Academy of Sciences*, 110(52):20935–20940, 2013. 101, 132

[181] F. R. Kschischang and B. J. Frey. Factor graphs and the sum-product algorithm. *Information Theory, IEEE*, 2001. 8

[182] H. W. Kuhn. The hungarian method for the assignment problem. *Naval research logistics quarterly*, 2 (1-2):83–97, 1955. 114

[183] X. Lan, S. Roth, D. Huttenlocher, and M. J. Black. Efficient Belief Propagation with Learned Higher-Order Markov Random Fields. *Computer VisionâĂŞECCV 2006*, 3952:269–282, 2006. 39

[184] A. Lancichinetti and S. Fortunato. Community detection algorithms: A comparative analysis. *Physical Review E*, 80(5):056117, 2009. 101

[185] S. L. Lauritzen and F. V. Jensen. Local computation with valuations from a commutative semigroup. *Annals of Mathematics and Artificial Intelligence*, 21(1):51–69, 1997. 2, 8

[186] S. L. Lauritzen and D. J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *Journal of the Royal Statistical Society Series B Methodological*, 50(2):157–224, 1988. 48

[187] N. Lazic. *Message Passing Algorithms for Facility Location Problems*. PhD thesis, University of Toronto, 2011. 103

[188] N. Lazic, B. J. Frey, and P. Aarabi. Solving the uncapacitated facility location problem using message passing algorithms. In *International Conference on Artificial Intelligence and Statistics*, pages 429–436, 2010. 103

[189] C. Lecoutre. A collection of csp benchmark instances., Oct. 2013. 78

[190] J. Leskovec, D. Chakrabarti, J. Kleinberg, C. Faloutsos, and Z. Ghahramani. Kronecker graphs: An approach to modeling networks. *The Journal of Machine Learning Research*, 11:985–1042, 2010. 132

[191] J. Leskovec, K. Lang, and M. Mahoney. Empirical comparison of algorithms for network community detection. In *WWW*, 2010. 101

[192] D. A. Levin, Y. Peres, and E. L. Wilmer. *Markov chains and mixing times*. American Mathematical Soc., 2009. 71

[193] S. Litsyn, R. E, and S. N. The table of nonlinear binary codes. http://www.eng.tau.ac.il/~litsyn/tableand/, 1999. [Online; accessed Jan 30 2014]. 96, 98

[194] M. L. Littman, S. M. Majercik, and T. Pitassi. Stochastic boolean satisfiability. *Journal of Automated Reasoning*, 27(3):251–296, 2001. 18, 21

[195] Q. Liu and A. Ihler. Variational algorithms for marginal map. *The Journal of Machine Learning Research*, 14(1):3165–3200, 2013. 47

[196] L. Lovász. Operations with structures. *Acta Mathematica Hungarica*, 18(3):321–328, 1967. 130

[197] L. Lovász and B. Szegedy. Limits of dense graph sequences. *Journal of Combinatorial Theory, Series B*, 96(6):933–957, 2006. 129

[198] B. D. MacArthur, R. J. Sánchez-García, and J. W. Anderson. Symmetry in complex networks. *Discrete Applied Mathematics*, 156(18):3525–3531, 2008. 126

[199] E. N. Maneva, E. Mossel, and M. J. Wainwright. A new look at survey propagation and its generalizations. *arXiv:cs/0409012*, Sept. 2004. 58

[200] C. D. Manning and H. Schütze. *Foundations of statistical natural language processing*. MIT press, 1999. 1

[201] S. Masuyama, T. Ibaraki, and T. Hasegawa. The computational complexity of the m-center problems on the plane. *IEICE TRANSACTIONS (1976-1990)*, 64(2):57–64, 1981. 105

[202] D. Maua and C. De Campos. Anytime marginal map inference. *arXiv preprint arXiv:1206.6424*, 2012. 47

[203] A. McCallum, A. Corrada-Emmanuel, and X. Wang. Topic and role discovery in social networks. *Computer Science Department Faculty Publication Series*, page 3, 2005. 1

[204] B. D. McKay. Small graphs are reconstructible. *Australasian Journal of Combinatorics*, 15:123–126, 1997. 131

[205] B. D. McKay and A. Piperno. Practical graph isomorphism, {II}. *Journal of Symbolic Computation*, 60 (0):94 – 112, 2014. 131

[206] O. Meshi, A. Jaimovich, A. Globerson, and N. Friedman. *Convexifying the Bethe free energy*. AUAI Press, June 2009. 46

[207] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller. Equations of state calculations by fast computing machines. *Journal of Medical Physics*, 21(6):1087–1092, 1953. 67

[208] M. Mézard and A. Montanari. *Information, physics, and computation*. Oxford, 2009. 1, 58, 64, 68, 71, 81, 86

[209] M. Mezard and G. Parisi. The bethe lattice spin glass revisited. *arXiv:cond-mat/0009418*, Sept. 2000. Eur. Phys. J. B 20, 217 (2001). 79

[210] M. Mezard and G. Parisi. The cavity method at zero temperature. *J. Stat. Phys 111 1*, 2003. 79, 80

[211] M. Mezard, G. Parisi, and M. A. Virasoro. *Spin Glass Theory and Beyond*. Singapore: World Scientific, 1987. 1, 46, 80

[212] M. Mezard, G. Parisi, M. A. Virasoro, and D. J. Thouless. Spin glass theory and beyond. *Physics Today*, 41:109, 1988. 13

[213] M. Mezard, G. Parisi, and R. Zecchina. Analytic and algorithmic solution of random satisfiability problems. *Science*, 297(5582):812–815, Aug. 2002. 2, 79

[214] T. Minka and Y. Qi. Tree-structured approximations by expectation propagation. In *NIPS*, 2003. 56

[215] D. Mitchell, B. Selman, and H. Levesque. Hard and easy distributions of sat problems. In *AAAI*, volume 92, pages 459–465, 1992. 2, 79

[216] R. Monasson, R. Zecchina, S. Kirkpatrick, B. Selman, and L. Troyansky. Determining computational complexity from characteristic âĂŸphase transitionsâĂŹ. *Nature*, 400(6740):133–137, 1999. 2

[217] A. Montanari and T. Rizzo. How to Compute Loop Corrections to Bethe Approximation. *Journal of Statistical Mechanics: Theory and Experiment*, (10):16, 2005. 50

[218] A. Montanari, F. Ricci-Tersenghi, and G. Semerjian. Solving constraint satisfaction problems through belief propagation-guided decimation. *arXiv:0709.1667*, Sept. 2007. 81

[219] A. Montanari, F. Ricci-Tersenghi, and G. Semerjian. Clusters of solutions and replica symmetry breaking in random k-satisfiability. *arXiv:0802.3627*, Feb. 2008. J. Stat. Mech. P04004 (2008). 87

[220] J. Mooij. *Understanding and improving belief propagation*. PhD thesis, Radboud Universiteit Nijmegen, 2008. 50, 52, 56

[221] J. Mooij. libDAI: A free and open source C++ library for discrete approximate inference in graphical models. *JMLR*, 2010. 56

[222] C. Moore and S. Mertens. *The nature of computation*. Oxford University Press, 2011. 1, 116, 125

[223] M.-L. Mugnier. Knowledge representation and reasonings based on graph homomorphism. In *Conceptual Structures: Logical, Linguistic, and Computational Issues*, pages 172–192. Springer, 2000. 129

[224] K. Murphy, S. Mian, et al. Modelling gene expression data using dynamic Bayesian networks. Technical report, Technical report, Computer Science Division, University of California, Berkeley, CA, 1999. 1

[225] K. Murphy, Y. Weiss, and M. Jordan. Loopy-belief Propagation for Approximate Inference: An Empirical Study. In *15*, pages 467–475, 1999. 36

[226] K. P. Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012. 132

[227] R. M. Neal. Probabilistic inference using markov chain monte carlo methods. 1993. 66

[228] M. Newman. Finding community structure in networks using the eigenvectors of matrices. *Physical review E*, 2006. 107, 112, 132

[229] M. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical Review E*, 2004. 107

[230] A. Y. Ng, M. I. Jordan, Y. Weiss, et al. On spectral clustering: Analysis and an algorithm. *Advances in neural information processing systems*, 2:849–856, 2002. 101

[231] N. Noorshams and M. J. Wainwright. Belief propagation for continuous state spaces: Stochastic message-passing with quantitative guarantees. *Journal of Machine Learning Research*, 14:2799–2835, 2013. 68

[232] L. E. Ortiz and M. Kearns. Nash propagation for loopy graphical games. In *Advances in Neural Information Processing Systems*, pages 793–800, 2002. 39

[233] M. Padberg and G. Rinaldi. A branch-and-cut algorithm for the resolution of large-scale symmetric traveling salesman problems. *SIAM review*, 33(1):60–100, 1991. 116

[234] R. Paget and I. D. Longstaff. Texture synthesis via a noncausal nonparametric multiscale Markov random field. *IEEE Transactions on Image Processing*, 7(6):925–931, 1998. 39

[235] P. Pakzad and V. Anantharam. Estimation and Marginalization Using the Kikuchi Approximation Methods. *Neural Computation*, 17(8):1836–1873, Aug. 2005. 49

[236] G. Palla, I. Derényi, I. Farkas, and T. Vicsek. Uncovering the overlapping community structure of complex networks in nature and society. *Nature*, 2005. 101

[237] R. Panigrahy and S. Vishwanathan. An o(log*n) approximation algorithm for the asymmetric p-center problem. *J Algorithms*, 1998. 29, 105

[238] C. H. Papadimitriou. The euclidean travelling salesman problem is np-complete. *Theoretical Computer Science*, 4(3):237–244, 1977. 116

[239] G. Papandreou and A. L. Yuille. Perturb-and-map random fields: Using discrete optimization to learn and sample from energy models. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 193–200. IEEE, 2011. 26

[240] G. Parisi. A backtracking survey propagation algorithm for k-satisfiability. *Arxiv preprint cond-mat0308510*, page 9, 2003. 81

[241] J. D. Park and A. Darwiche. Complexity results and approximation strategies for map explanations. *J. Artif. Intell. Res.(JAIR)*, 21:101–133, 2004. 20, 38

[242] R. G. Parker and R. L. Rardin. Guaranteed performance heuristics for the bottleneck travelling salesman problem. *Operations Research Letters*, 2(6):269–272, 1984. 29, 122

[243] J. Pearl. Bayesian networks: a model of self-activated memory for evidential reasoning. 1985. 8

[244] J. Pearl. *Probabilistic reasoning in intelligent systems*. 1988. 1

[245] J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*, volume 88 of *Representation and Reasoning*. Morgan Kaufmann, 1988. 2, 49

[246] A. Pelizzola. Cluster Variation Method in Statistical Physics and Probabilistic Graphical Models. *Journal of Physics A: Mathematical and General*, 38(33):36, 2005. 48

[247] C. C. Pinter. *A book of abstract algebra*. Courier Dover Publications, 2012. 9, 24

[248] P. Pons and M. Latapy. Computing communities in large networks using random walks. In *Computer and Information Sciences-ISCIS 2005*, pages 284–293. Springer, 2005. 101

[249] B. Potetz and T. S. Lee. Efficient belief propagation for higher-order cliques using linear constraint nodes. *Computer Vision and Image Understanding*, 112(1):39–54, 2008. 39, 40

[250] C. Pralet, G. Verfaillie, and T. Schiex. An algebraic graphical model for decision with uncertainties, feasibilities, and utilities. *J. Artif. Intell. Res.(JAIR)*, 29:421–489, 2007. 9

[251] U. N. Raghavan, R. Albert, and S. Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Physical Review E*, 76(3):036106, 2007. 101

[252] A. Ramezanpour and R. Zecchina. Cavity approach to sphere packing in hamming space. *Physical Review E*, 85(2):021106, 2012. 95

[253] S. Ravanbakhsh and R. Greiner. Perturbed message passing for constraint satisfaction problems. *JMLR, to appear*, 2014. 4

[254] S. Ravanbakhsh, R. Greiner, and J. Y. Chun-nam. A generalized loop correction method for approximate inference in graphical models. In *Proceedings of the 29th International Conference on Machine Learning (ICML-12)*, pages 543–550, 2012. 4, 50, 55

[255] S. Ravanbakhsh, P. Liu, R. Mandal, J. R. Grant, M. Wilson, R. Eisner, I. Sinelnikov, X. Hu, C. Luchinat, R. Greiner, et al. Accurate, fully-automated nmr spectral profiling for metabolomics. *arXiv preprint arXiv:1409.1456*, 2014. 1

[256] S. Ravanbakhsh, R. Rabbany, and R. Greiner. Revisiting algebra and complexity of inference in graphical models. *arXiv:1409.7410*, 2014. 4

[257] S. Ravanbakhsh, R. Rabbany, and R. Greiner. Augmentative message passing for traveling salesman problem and graph partitioning. *NIPS*, 2014. 4

[258] S. Ravanbakhsh, C. Srinivasa, B. Frey, and R. Greiner. Min-max problems on factor-graphs. *Proceedings of T 31st International Conference on Machine Learning, pp. 1035-1043*, 2014. 4, 17

[259] J. Reichardt and S. Bornholdt. Detecting fuzzy community structures in complex networks with a potts model. *Physical Review Letters*, 93(21):218701, 2004. 107, 112

[260] G. Reinelt. TsplibâĂŤa traveling salesman problem library. *ORSA journal on computing*, 3(4):376–384, 1991. 119

[261] F. Ricci-Tersenghi and G. Semerjian. On the cavity method for decimated random constraint satisfaction problems and the analysis of belief propagation guided decimation algorithms. *arXiv:0904.3395*, Apr. 2009. J Stat. Mech. P09001 (2009). 81

[262] T. Rizzo, B. Wemmenhove, and H. J. Kappen. On Cavity Approximations for Graphical Models. *Physical Review E*, 76(1):1–9, 2006. 50

[263] C. P. Robert and G. Casella. *Monte Carlo Statistical Methods (Springer Texts in Statistics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2005. 66

[264] H. Rogers. Theory of recursive functions and effective computability. 1987. 13

[265] P. Ronhovde and Z. Nussinov. Local resolution-limit-free potts model for community detection. *Physical Review E*, 81(4):046114, 2010. 107

[266] M. Rosvall and C. T. Bergstrom. Maps of random walks on complex networks reveal community structure. *PNAS*, 2008. 101

[267] D. Roth. On the hardness of approximate reasoning. In *Proceedings of the 13th international joint conference on Artifical intelligence-Volume 1*, pages 613–618. Morgan Kaufmann Publishers Inc., 1993. 2, 18

[268] S. Roth and M. J. Black. Fields of Experts. *International Journal of Computer Vision*, 82(2):205–229, 2009. 39

[269] C. Rother and P. Kohli. Minimizing sparse higher order energy functions of discrete variables. *IEEE Conference on Computer Vision and Pattern Recognition (2009)*, 0:1382–1389, 2009. 39

[270] O. Roussel and C. Lecoutre. Xml representation of constraint networks: Format xcsp 2.1. *arXiv preprint arXiv:0902.2362*, 2009. 78

[271] N. Ruozzi and S. Tatikonda. Message-passing algorithms: Reparameterizations and splittings. *Information Theory, IEEE Transactions on*, 59(9):5860–5881, 2013. 46

[272] A. Saade, F. Krzakala, and L. Zdeborová. Spectral clustering of graphs with the bethe hessian. *arXiv preprint arXiv:1406.1880*, 2014. 132

[273] S. Sanghavi. Equivalence of lp relaxation and max-product for weighted matching in general graphs. In *Information Theory Workshop, 2007. ITW'07. IEEE*, pages 242–247. IEEE, 2007. 116

[274] S. Sanghavi, D. Shah, and A. S. Willsky. Message passing for max-weight independent set. In *Advances in Neural Information Processing Systems*, pages 1281–1288, 2007. 26, 99

[275] T. Schiex, H. Fargier, and G. Verfaillie. Valued constraint satisfaction problems: hard and easy problems. In *Proceedings of the 14th international joint conference on Artificial intelligence-Volume 1*, pages 631–637. Morgan Kaufmann Publishers Inc., 1995. 8

[276] A. Schrijver. *Min-max results in combinatorial optimization*. Springer, 1983. 29

[277] B. Selman, H. A. Kautz, and B. Cohen. Noise strategies for improving local search. In *AAAI*, volume 94, pages 337–343, 1994. 79, 80

[278] P. P. Shenoy. Valuation-based systems: A framework for managing uncertainty in expert systems. In *Fuzzy logic for the management of uncertainty*, pages 83–104. John Wiley & Sons, Inc., 1992. 8

[279] P. P. Shenoy and G. Shafer. Axioms for probability and belief-function propagation. In *Uncertainty in Artificial Intelligence*, 1990. 2

[280] D. Sherrington and S. Kirkpatrick. Solvable model of a spin-glass. *Physical review letters*, 35(26):1792, 1975. 13

[281] S. E. Shimony. Finding maps for belief networks is np-hard. *Artificial Intelligence*, 68(2):399–410, 1994. 17

[282] L. Song, A. Gretton, and C. Guestrin. Nonparametric Tree Graphical Models. *AISTATS 13*, pages 765–772, 2010. 68

[283] L. Song, A. Gretton, D. Bickson, Y. Low, and C. Guestrin. Kernel Belief Propagation. *Artificial Intelligence*, 15(m):11–13, 2011. 68

[284] D. Sontag and T. S. Jaakkola. New outer bounds on the marginal polytope. In *Advances in Neural Information Processing Systems*, pages 1393–1400, 2007. 42

[285] D. Sontag, T. Meltzer, A. Globerson, T. S. Jaakkola, and Y. Weiss. Tightening lp relaxations for map using message passing. *arXiv preprint arXiv:1206.3288*, 2012. 46

[286] D. A. Sontag. *Cutting plane algorithms for variational inference in graphical models*. PhD thesis, Massachusetts Institute of Technology, 2007. 42

[287] K. Steiglitz and C. H. Papadimitriou. Combinatorial optimization: Algorithms and complexity. *Printice-Hall, New Jersey*, 1982. 1

[288] L. J. Stockmeyer. The polynomial-time hierarchy. *Theoretical Computer Science*, 3(1):1–22, 1976. 13

[289] J. X. J. W. P. Tan and L. Quan. Joint affinity propagation for multiple view segmentation. 103

[290] R. M. Tanner. A recursive approach to low complexity codes. *Information Theory, IEEE Transactions on*, 27(5):533–547, 1981. 1

[291] D. Tarlow, I. E. Givoni, and R. S. Zemel. Hop-map: Efficient message passing with high order potentials. In *International Conference on Artificial Intelligence and Statistics*, pages 812–819, 2010. 39

[292] D. Tarlow, K. Swersky, R. S. Zemel, R. P. Adams, and B. J. Frey. Fast exact inference for recursive cardinality models. *arXiv preprint arXiv:1210.4899*, 2012. 39, 40

[293] Y. W. Teh and M. Welling. The unified propagation and scaling algorithm. *Advances in Neural Information Processing {...}*, 2002. 46

[294] S. Thrun. Particle filters in robotics. In *Proceedings of the Eighteenth conference on Uncertainty in artificial intelligence*, pages 511–518. Morgan Kaufmann Publishers Inc., 2002. 1

[295] L. Tierney. Markov chains for exploring posterior distributions. *Annals of Statistics*, 22(4):1701–1728, 1994. 68

[296] S. Toda. Pp is as hard as the polynomial-time hierarchy. *SIAM Journal on Computing*, 20(5):865–877, 1991. 21

[297] L. G. Valiant. The complexity of computing the permanent. *Theoretical computer science*, 8(2):189–201, 1979. 18, 113

[298] L. G. Valiant and V. V. Vazirani. Np is as easy as detecting unique solutions. *Theoretical Computer Science*, 47:85–93, 1986. 25

[299] U. Von Luxburg. A tutorial on spectral clustering. *Statistics and computing*, 17(4):395–416, 2007. 132

[300] J. Von Neumann and O. Morgenstern. *Theory of Games and Economic Behavior (60th Anniversary Commemorative Edition)*. Princeton university press, 2007. 39

[301] P. O. Vontobel. The bethe permanent of a nonnegative matrix. *Information Theory, IEEE Transactions on*, 59(3):1866–1901, 2013. 114

[302] K. W. Wagner. The complexity of combinatorial problems with succinct input representation. *Acta Informatica*, 23(3):325–356, 1986. 13

[303] M. J. Wainwright and M. I. Jordan. Graphical Models, Exponential Families, and Variational Inference. *Foundations and TrendsÂő in Machine Learning*, 1(1âĂŞ2):1–305, 2007. 1, 43

[304] M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. A New Class of Upper Bounds on the Log Partition Function. *IEEE Transactions on Information Theory*, 51(7):2313–2335, July 2005. 46

[305] M. J. Wainwright, T. S. Jaakkola, and A. S. Willsky. MAP Estimation Via Agreement on Trees: Message-Passing and Linear Programming. *IEEE Transactions on Information Theory*, 51(11):3697–3717, Nov. 2005. 46

[306] C. Wang, J. Lai, and W. Zheng. Message-passing for the traveling salesman problem. 122

[307] S. Wasserman. *Social network analysis: Methods and applications*, volume 8. Cambridge university press, 1994. 1

[308] Y. Watanabe and M. Chertkov. Belief propagation and loop calculus for the permanent of a non-negative matrix. *Journal of Physics A: Mathematical and Theoretical*, 43(24):242002, 2010. 114

[309] M. Weigt and H. Zhou. Message passing for vertex covers. *Physical Review E*, 74(4):046110, 2006. 100

[310] Y. Weiss. Correctness of belief propagation in Gaussian graphical models of arbitrary topology. *Neural computation*, (June), 2001. 36

[311] Y. Weiss, C. Yanover, and T. Meltzer. MAP Estimation, Linear Programming and Belief Propagation with Convex Free Energies. *Uncertainty in Artificial Intelligence*, pages 416–425, 2007. 46

[312] A. Weller and T. Jebara. Clamping variables and approximate inference. In *Advances in Neural Information Processing Systems*, pages 909–917, 2014. 49

[313] A. Weller and T. S. Jebara. On map inference by mwss on perfect graphs. *arXiv preprint arXiv:1309.6872*, 2013. 36

[314] M. Welling. On the choice of regions for generalized belief propagation. In *Proceedings of the 20th conference on Uncertainty in {...}*, 2004. 49

[315] M. Welling. A Cluster-Cumulant Expansion at the Fixed Points of Belief Propagation. *UAI*, (x), 2012. 50

[316] M. Welling, M. Rosen-Zvi, and G. E. Hinton. Exponential family harmoniums with an application to information retrieval. In *Advances in neural information processing systems*, pages 1481–1488, 2004. 1

[317] M. Welling, T. Minka, and Y. Teh. Structured Region Graphs: Morphing EP into GBP. *UAI 2005*, (1998):609–616, 2005. 49

[318] S. White and P. Smyth. A spectral clustering approach to finding communities in graph. In *SDM*. SIAM, 2005. 101

[319] W. Wiegerinck, T. Heskes, et al. Fractional belief propagation. *Advances in Neural Information Processing Systems*, pages 455–462, 2003. 46

[320] J.-Q. Xiao and H. Zhou. Partition function loop series for a general graphical model: free energy corrections and message-passing equations. *Journal of Physics A: Mathematical and Theoretical*, 44 (42):12, 2011. 50

[321] Y. Xiao, M. Xiong, W. Wang, and H. Wang. Emergence of symmetry in complex networks. *Physical Review E*, 77(6):066108, 2008. 126

[322] X. Xu, N. Yuruk, Z. Feng, and T. A. Schweiger. Scan: a structural clustering algorithm for networks. In *SIGKDD*, 2007. 101

[323] C. Yanover and Y. Weiss. Approximate inference and protein-folding. In *Advances in neural information processing systems*, pages 1457–1464, 2002. 1

[324] C. Yanover, T. Meltzer, and Y. Weiss. Linear Programming Relaxations and Belief Propagation - an Empirical Study. *Journal of Machine Learning Research*, 7:1887–1907, 2006. 46

[325] C. Yanover, T. Meltzer, and Y. Weiss. Linear programming relaxations and belief propagation–an empirical study. *JMLR*, 2006. 42

[326] J. Yedidia and W. Freeman. Generalized belief propagation. *Advances in neural information*, 2001. 2

[327] J. Yedidia, W. Freeman, and Y. Weiss. Constructing Free-Energy Approximations and Generalized Belief Propagation Algorithms. *IEEE Transactions on Information Theory*, 51(7):2282–2312, 2005. 45, 49

[328] J. S. Yedidia, W. T. Freeman, and Y. Weiss. Characterization of belief propagation and its generalizations. *ITIEEE*, 51(TR-2001-15):1–16, 2001. 45, 48, 49

[329] A. L. Yuille. CCCP algorithms to minimize the Bethe and Kikuchi free energies: convergent alternatives to belief propagation. *Neural Computation*, 14(7):1691–1722, 2002. 46

[330] L. Zdeborova and F. Krzakala. Phase transitions in the coloring of random graphs. *Physical Review E*, 76(3):031131, Sept. 2007. 87

[331] L. Zdeborová and M. Mézard. The number of matchings in random graphs. *Journal of Statistical Mechanics: Theory and Experiment*, 2006(05):P05003, 2006. 115

[332] P. Zhang and C. Moore. Scalable detection of statistically significant communities and hierarchies: message-passing for modularity. *arXiv preprint arXiv:1403.5787*, 2014. 107

[333] H. Zhou. Vertex cover problem studied by cavity method: Analytics and population dynamics. *The European Physical Journal B-Condensed Matter and Complex Systems*, 32(2):265–270, 2003. 100

[334] H. Zhou, C. Wang, J.-Q. Xiao, and Z. Bi. Partition function expansion on region graphs and message-passing equations. *Journal of Statistical Mechanics: Theory and Experiment*, 2011(12):L12001, Dec. 2011. 50

Appendix

input : Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with normalized (weighted)adjacency $\mathbf{A}$, maximum iterations $T_{\max}$, damping λ, threshold $\epsilon_{\max}$.
output: A clustering $C = \{C_1, \ldots, C_K\}$ of nodes.
construct the null model
$\widehat{\mathrm{p}}_{i:j} \leftarrow 0 \; \forall (i,j) \in \mathcal{E} \cup \mathcal{E}^{\text{NULL}}$
while *TRUE* **do** `// the augmentation loop`

- $\epsilon \leftarrow 0, T \leftarrow 0$
- **while** $\epsilon < \epsilon_{\max}$ *and* $T < T_{\max}$ **do** `// BP loop`
  - $\epsilon \leftarrow 0$
  - **for** $(i,j) \in \mathcal{E} \cup \mathcal{E}^{NULL}$ **do**
    - $\widehat{\mathrm{p}}'_{i:j} \leftarrow \widehat{\mathrm{p}}_{i:j}$
    - $\widehat{\mathrm{p}}_{i:j} \leftarrow (\mathbf{A}_{i,j} - \mathbf{A}^{\text{NULL}}_{i,j})$
    - **for** $\mathrm{I} \in \partial i : j$ **do** `// update beliefs`
      - calculate $\widehat{\mathrm{p}}_{\mathrm{I} \to i:j}$ using equation (4.7)
      - $\widehat{\mathrm{p}}_{i:j} \leftarrow \widehat{\mathrm{p}}_{i:j} + \widehat{\mathrm{p}}_{\mathrm{I} \to i:j}$
    - **end**
    - $\epsilon \leftarrow \max\{\epsilon, |\widehat{\mathrm{p}}_{i:j} - \widehat{\mathrm{p}}'_{i:j}|\}$
    - **for** $\mathrm{I} \in \partial i : j$ **do** `// update msgs.`
      - $\widetilde{\mathrm{p}}_{i:j \to \mathrm{I}} \leftarrow \widehat{\mathrm{p}}_{i:j} - \widehat{\mathrm{p}}_{\mathrm{I} \to i:j}$
      - $\widehat{\mathrm{p}}_{i:j \to \mathrm{I}} \leftarrow \lambda \widetilde{\mathrm{p}}_{i:j \to \mathrm{I}} + (1 - \lambda) \widehat{\mathrm{p}}_{i:j \to \mathrm{I}}$
    - **end**
  - **end**
  - $T \leftarrow T + 1$
- **end**
- **for** $i \in \mathcal{V}$ **do**
  - **for** $(i,j), (i,k) \in \mathcal{E} \cup \mathcal{E}^{NULL}$ **do**
    - **if** $\widehat{\mathrm{p}}_{i:j} > 0$ ***and*** $\widehat{\mathrm{p}}_{i:k} > 0$ ***and*** $\widehat{\mathrm{p}}_{i:k} \leq 0$ **then** add the corresponding clique factor to the factor-graph
  - **end**
- **end**
- **if** *no factor was added* **then break** out of the loop
- **else** $\widehat{\mathrm{p}}_{i:j \to \mathrm{I}} \leftarrow 0 \; \forall \mathrm{I}, i : j \in \mathrm{I}$

end
$C \leftarrow \texttt{ConnectedComponents}((\mathcal{V}, \{(i,j) \in \mathcal{E} \cup \mathcal{E}^{NULL} \mid \widehat{\mathrm{p}}_{i:j} > 0\}))$

Algorithm 3: Message Passing for Modularity Maximization.

input : Graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, weighted (symmetric) adjacency matrix $\mathbf{A}$, maximum iterations $T_{\max}$, damping λ, threshold $\epsilon_{\max}$.
output: A subset $\mathcal{T} \subset \mathcal{E}$ of the edges in the tour.
construct the initial factor-graph
initialize the messages for degree constraints $\widehat{\mathrm{p}}_{i:j \to \mathcal{E}(i,\cdot)} \leftarrow 0 \forall i \in \mathcal{V}, j \in \mathcal{E}(i,\cdot)$
initialize $\widehat{\mathrm{p}}_{i:j} \leftarrow \mathbf{A}_{i,j} \;\; \forall (i,j) \in \mathcal{E}$
while *TRUE* **do** `// the augmentation loop`

$\epsilon \leftarrow 0, T \leftarrow 0$
while $\epsilon < \epsilon_{\max}$ ***and*** $T < T_{\max}$ **do** `// BP loop`

$\epsilon \leftarrow 0$
for ***each*** $\mathrm{f_I}$ **do** `// including` $\mathrm{f}_{\mathcal{E}(\mathcal{S},\cdot)}$, $\mathrm{f}_{\mathcal{E}(i,\cdot)}$ `(updates all the outgoing messages from this factor)`

find three lowest values in $\{\widehat{\mathrm{p}}_{i:j \to \mathrm{I}} \mid i:j \in \partial \mathrm{I}\}$
for ***each*** $i:j \in \mathrm{I}$ **do**
calculate $\widetilde{\mathrm{p}}_{\mathrm{I} \to i:j}$ using equation (5.7)
$\epsilon_{\mathrm{I} \to i:j} \leftarrow \widetilde{\mathrm{p}}_{\mathrm{I} \to i:j} - \widehat{\mathrm{p}}_{\mathrm{I} \to i:j}$
$\widehat{\mathrm{p}}_{\mathrm{I} \to i:j} \leftarrow \widehat{\mathrm{p}}_{\mathrm{I} \to i:j} + \lambda \epsilon_{\mathrm{I} \to i:j}$
$\widehat{\mathrm{p}}_{i:j} \leftarrow \widehat{\mathrm{p}}_{i:j} + \epsilon_{\mathrm{I} \to i:j}$
$\epsilon \leftarrow \max(\epsilon, |\epsilon_{\mathrm{I} \to e}|)$
end
end
$T \leftarrow T + 1$
end
$\mathcal{T} \leftarrow \{(i,j) \in \mathcal{E} \mid \widehat{\mathrm{p}}_{i:j} > 0\}$ `// respecting degree constraints.`
$\mathcal{C} \leftarrow$ `ConnectedComponents`$((\mathcal{V}, \mathcal{T}))$
if $|\mathcal{C}| = 1$ **then** return $\mathcal{T}$
else augment the factor-graph with $\mathrm{f}_{\mathcal{E}(\mathcal{S},\cdot)} \; \forall \mathcal{S} \in \mathcal{C}$
initialize $\widehat{\mathrm{p}}_{\mathcal{E}(\mathcal{S},\cdot) \to i:j} \leftarrow 0 \; \forall \mathcal{S} \in \mathcal{C}, i:j \in \mathcal{E}(\mathcal{S},\cdot)$

end

Algorithm 4: Message Passing for TSP