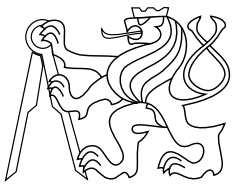
CENTER FOR
MACHINE PERCEPTION



CZECH TECHNICAL
UNIVERSITY IN PRAGUE

HABILITATION THESIS

Approximate Inference in Graphical Models

Tomáš Werner

October 20, 2014

Center for Machine Perception, Department of Cybernetics
Faculty of Electrical Engineering, Czech Technical University
Technická 2, 166 27 Prague 6, Czech Republic
fax +420 2 2435 7385, phone +420 2 2435 7637, www: http://cmp.felk.cvut.cz

Contents

Chapter 1

Introduction

Graphical models combine graph theory and probability theory to a general formalism to model interactions of a set of variables. The formalism finds many applications in pattern recognition, artificial intelligence, computer vision, and other disciplines. A basic operation with a graphical model is *inference*, which requires computation of either the maximum or the marginals of the probability distribution defined by the model. Up to rare cases, computing these quantities is intractable and one has to recourse to approximative algorithms.

This thesis documents my contributions to approximate inference in graphical models. My baseline was an old and widely unknown (but recently rediscovered) approach to maximizing the distribution by Schlesinger et al., based on linear programming relaxation. I generalized this approach, which enabled handling interactions of arbitrary arity, constructing a hierarchy of progressively tighter relaxations, and incrementally tightening the relaxation in a cutting plane fashion. Then I generalized the resulting message-massing algorithm in yet another way, based on the concept of commutative semiring. This unified it with several other algorithms, in particular constraint propagation widely used in constraint programming. Finally, I used insights obtained from the LP relaxation framework to derive an alternative view on the dynamics of loopy belief propagation, a famous algorithm to approximate marginals of a graphical model.

The thesis has the form of a collection of selected publications, endowed with an introductory text. Chapter 2 gives background on graphical models and in Chapter 3 surveys our novel contributions. Then a collection of publications, which were referred to in Chapter 3, follows.

Chapter 2

Background

In this chapter, we review the formalism of graphical models and their applications. Then we define the two problems needed to make inference in graphical models, the sum-product problem and the max-product problem.

2.1 Graphical Models

The most general way how to model interaction of a set of variables is by specifying their joint probability distribution. Unless the number of variables is very small, such a distribution in its unrestricted form is impossible to represent in the computer or estimate from data. This 'curse of dimensionality' can be avoided by restricting the form of the distribution to be a product of functions, each depending on only a subset of the variables. Formally, let our set of variables be V, where a variable $v \in V$ attains states x_v from some finite domain X_v. Let $E \subseteq 2^V$ be a system of variable subsets, thus (V, E) can be understood as a hypergraph and $A \in E$ as a hyperedge. The joint distribution then has the form

$$p(x_V) = \frac{1}{Z(f)} \prod_{A \in E} f_A(x_A) \tag{2.1}$$

where $x_A = (\, x_v \mid v \in A \,)$ denotes the joint state of subset $A \subseteq V$ of the variables and each f_A is a function that maps x_A to a non-negative number. We assume the variable domains X_v to be finite, thus f_A is represented by an $|A|$-dimensional array. The distribution is normalized by the *partition function*

$$Z(f) = \sum_{x_V} \prod_{A \in E} f_A(x_A). \tag{2.2}$$

For example, let $V = (1, 2, 3, 4)$ and $E = \{(1, 2), (1, 3), (2, 3, 4)\}$. Then

$$p(x_V) = p(x_1, x_2, x_3, x_4) = \frac{1}{Z(f)}\, f_{12}(x_1, x_2)\, f_{13}(x_1, x_3)\, f_{234}(x_2, x_3, x_4)$$

where

$$Z(f) = \sum_{x_1, x_2, x_3, x_4} f_{12}(x_1, x_2)\, f_{13}(x_1, x_3)\, f_{234}(x_2, x_3, x_4).$$

Suppose now that each variable has 3 states, $|X_v| = 3$ for all v. Then the number of all possible functions $p(x_V)$ in unrestricted form is $3^4 = 81$, whereas the number of all possible functions in the form (2.1) is only $3^2 + 3^2 + 3^3 = 45$.

We have just described the basic idea behind *graphical models* (Lauritzen, 1996; Bishop, 2006; Wainwright and Jordan, 2008). Two types of graphical models exist, directed and undirected. Distribution (2.1) is known as the *Gibbs distribution*, modeled by an undirected graphical model known as *Gibbs random field* (which is slightly more general than more widely known *Markov random field*). In the thesis, we focus on graphical models in this very general form.

The formalism of graphical models can be applied in a remarkably wide range of settings, see e.g. (Wainwright and Jordan, 2008, section 2.4). Acyclic graphical models (*Markov chains* or *Markov models*) have proven extremely successful in speech recognition and bioinformatics (Durbin et al., 1999). General graphical models have been applied to bioinformatics (Kingsford et al., 2005; Sanchez et al., 2008), artificial intelligence, combinatorial optimization, economics, optimal assignment of radio link frequencies (Koster et al., 1998; Aardal et al., 2007), internet data mining, and error-correcting codes (Wainwright and Jordan, 2008, §2.4.7). In image analysis, they have been used in denoising, texture modeling, segmentation, 3-dimensional reconstruction (incl. segmentation in 3-D and space carving), in-painting, recognition, image registration, and optical flow. In every recent conference on computer vision or machine learning, several new contributions on graphical models appear.

A strong feature of graphical models is *multidisciplinarity* – they are subject to research in at least the following disciplines:

- *Statistical physics* (Mézard and Montanari, 2009; Mézard, 2003) aims to understand how macroscopic properties of matter result from random behavior of locally interacting particles. A famous example is the explanation of ferromagnetism from interaction of spin orientations by the *Ising model*.
- *Pattern recognition, machine learning, and computer vision* (Lauritzen, 1996; Koller and Friedman, 2009; Bishop, 2006; Wainwright and Jordan, 2008) aim at developing algorithms for intelligent processing of noisy information on a computer, often motivated by similar abilities of living organisms.
- *Constraint programming* (Rossi et al., 2006) is motivated by the idea that rather than to specify *how* to fulfill the task at hand (by giving a code), it is often more convenient to specify *what* should be fulfilled (by giving a set of constraints). The core of constraint programming is the *constraint satisfaction problem*, which seeks to satisfy given relations over given subsets of variables. Recently, this (crisp) constraint satisfaction has been extended to the *weighted* constraint satisfaction (Rossi et al., 2006, chapter 9), which is isomorphic to our maximization problem.

Vast literature on graphical models exists within each discipline but transfer of results across discipline borders is hindered by different terminologies and backgrounds. Though each discipline focuses on different aspects of graphical models, there is one aspect in common: modeling a (complex) global behavior of a system by (simple) local interactions of its parts. Arguably, this *emergence of global from local* is most interesting and profound about graphical models.

2.2 Inference in Graphical Models

Inference in a graphical model means to compute the states of a subset of the variables V (*hidden* variables) from the states of the remaining variables (*observed* variables). According to the Bayesian decision theory, this is formal-

ized as minimizing the expectation of a given loss function, which specifies the penalty for incorrect decisions. Two loss functions are most common:

- One results in maximizing the marginal distribution of each hidden variable separately (*maximum posterior marginal* or *MPM* inference). Then the inference algorithms require computing expressions of the type

$$\sum_{x_V} \prod_{A \in E} f_A(x_A), \tag{2.3}$$

  which is known as the *max-product problem*.

- The other results in maximizing the joint distribution over all hidden variables (*maximum a posteriori* or *MAP* inference). It follows that the inference algorithms require computing expressions of the type

$$\max_{x_V} \prod_{A \in E} f_A(x_A), \tag{2.4}$$

  which is known as the *max-product problem*.

The sets V, E and the functions f_A in (2.4) and (2.3) are not the same as in (2.1) because we sum or maximize only over the hidden variables. We adopted this simplification to avoid extra notation.

Calculating expressions (2.4) and (2.3) is intractable, requiring to sum or maximize over all states x_V of all the variables (an exponential number of elements). The most notable tractable subclass is formed by the problems with an *acyclic structure* (the factor graph defined by E is a tree), which can be exactly solved by *dynamic programming* and related algorithms (Aji and McEliece, 2000). Otherwise, we have to recourse to *approximate inference*. Algorithms for approximate inference underwent three revolutions in the past (the last one continues still today):

- In *Gibbs sampling* (Geman and Geman, 1984), samples are drawn from the distribution by local operations and accumulating these samples yields estimates of (max-)marginals.

- *Loopy belief propagation* (Pearl, 1988). Belief propagation is a finite algorithm, similar to dynamic programming, to compute (max-)marginals of distribution (2.1) on an acyclic (hyper)graph E. If applied to a *cyclic* (hyper)graph (which is straightforward), it often converges to a fixed point which yields surprisingly good approximations of (max-)marginals for many practical problems. The reason of this behavior is not well understood.

- Algorithms based on *linear programming (LP) relaxation* (Shlezinger, 1976; Werner, 2007b; Kolmogorov, 2006; Wainwright and Jordan, 2008). The max-product problem can be cast as an integer linear programming, the relaxation of which is tight for a large class of problems, much larger than just acyclic ones. This class includes also *supermodular* problems (Schlesinger and Flach, 2000; Werner, 2007b), which can be solved very efficiently by max-flow algorithms (Kolmogorov and Zabih, 2002; Schlesinger and Flach, 2006).

It is interesting that despite its very natural formulation, the max-product problem in its full generality has not been studied in mathematical optimization, only its subproblems such as the max-cut problem (Deza and Laurent, 1997) have been studied.

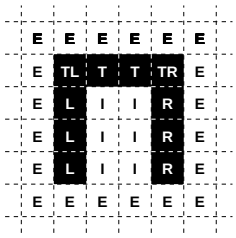
2.3 Example

To get the flavor of inference algorithms, let our distribution model the class of images containing non-overlapping letters 'Π' of arbitrary width and height, an example is the right-most image below. Let V be the set of pixels and let $E \subseteq \binom{V}{2}$ be the set of pairs of neighboring pixels. Each variable v attains states x_v from the set $X_v = \{\mathsf{E}, \mathsf{I}, \mathsf{T}, \mathsf{L}, \mathsf{R}, \mathsf{TL}, \mathsf{TR}\}$, representing syntactic parts of the letter Π. Let $f_v(y_v \,|\, x_v)$ be the probability that pixel v has color y_v given that it has state x_v. Let $f_{uv}(x_u, x_v)$ be 1 if syntactic parts x_u and x_v are ever incident (in the left picture below) and 0 otherwise. Then

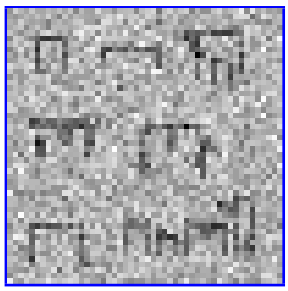
$$p(x_V, y_V) = \frac{1}{Z(f)} \underbrace{\prod_{v \in V} f_v(y_v \,|\, x_v)}_{\text{data}} \underbrace{\prod_{uv \in E} f_{uv}(x_u, x_v)}_{\text{prior}} \tag{2.5}$$

is the joint probability of an image x_V formed by the syntactic parts (hidden variables) and an input image y_V (observed variables). Maximizing $p(x_V, y_V)$ over x_V yields the most probable image from the class, given the input image y_V. Note that the normalization factor $Z(f)$ can be omitted in the maximization.

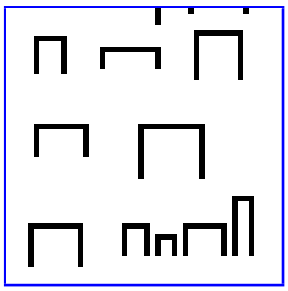
Maximization is done by iterating a simple local operation, which converges to an optimum of the dual LP relaxation (Shlezinger, 1976; Werner, 2007b). Despite the problem is neither acyclic nor supermodular, its LP relaxation is often tight (the exact maximum is been found). This allows us to 'parse' even very noisy images to its syntactic parts, as seen on the figure. Note that the linear program we solve is large-scale – in case of large images it can easily have millions of variables. A physicist could see this as 'statistical mechanics of images': complex global patterns emerge from simple local interactions f_{uv}.



syntactic parts
of letter 'Π'



input image y_V



output image x_V

Chapter 3

Contributions

This chapter surveys my contributions to approximate inference in graphical models in the form (2.1). The contributions are divided into four groups, described in the four sections below. For each group, all relevant publications are cited, of which a subset is selected, which is indicated at the end of each section. These selected publication are included as part of the thesis.

3.1 Review of the LP Relaxation Approach

My interest in graphical models began around the year 2000 when I attended several lectures given by Mikhail I. Schlesinger from the Glushkov Institute of Cybernetics in Kiev, Ukraine, who was visiting our department that time. It turned out that he formulated the basics of the LP relaxation approach to the max-product problem as early as in 1970's (Shlezinger, 1976). The approach consisted in minimizing an upper bound on the true (intractable) maximum (2.4) by linear transformations of (the logarithm of) f that preserve the distribution (2.1). This leads to a linear program, which in fact was dual to the LP relaxation of the original problem. Schlesinger and his colleagues proposed algorithms to minimize the upper bound: a very simple distributed ('message passing') algorithm nicknamed 'max-sum diffusion' (Kovalevsky and Koval, approx. 1975) and the algorithm (Koval and Schlesinger, 1976).

I became attracted by the topic and wanted to contribute to it. However, the approach was not known in computer vision and related fields, and even the current work by Schlesinger and Flach (2000) was somewhat disconnected from the main-stream literature. In fact, active knowledge of optimization of a typical computer vision researcher that time rarely reached beyond the Levenberg-Marquardt algorithm. Having worked on multiple view geometry before, I was no exception and had a hard time to understand the approach. Therefore, before attempting any novel contribution, I decided to first write a review (Werner, 2005, 2007b) which would summarize the approach of Schlesinger et al. in the modern terminology, relate it to the current state of the art, and re-implement the algorithms. The topic turned out to be very multifaceted and I found lots of relevant papers from quite different fields, many of them using different formulations and terminology. Interestingly, it further turned out that a similar approach was independently discovered approximately at the same time by Wainwright et al. (2005) and Kolmogorov (2006).

Publications included in the thesis: (Werner, 2007b)

3.2 Interactions of Arbitrary Arity

The *arity* of an interaction f_A is the number of variables on which it depends, that is, $|A|$. The LP relaxation approach described above was formulated only for interactions of arity at most 2 (often called 'binary', 'pairwise', or 'second-order' interactions), such as in (2.5). Seemingly, this is without any loss of generality because any non-binary interaction can be represented by combining binary interactions. However, representing some high-arity interactions need an exponential number of binary interactions. Moreover, this translation destroys symmetry. Therefore, I wanted to generalize the approach to handle interactions of arbitrary arity *natively*, without translating them to binary interactions and without sacrificing symmetry.

First I derived the LP relaxation approach for a very general family of distributions known as the (discrete) exponential family (Wainwright and Jordan, 2008), which contains distributions of type (2.1) but also other distributions. This is described in the technical report (Werner, 2009). Most of this report was written in fact much earlier than in 2009.

Then I took a less general view, considering only distributions of the form (2.1). This resulted in a surprisingly elegant framework with several desirable properties:

- It is applicable to interactions of arbitrary arity but it is at the same time very simple. The algorithm is a direct extension of max-sum diffusion as I formulated it in (Werner, 2007b).
- It straightforwardly allows to construct a partially-ordered hierarchy of progressively tighter LP relaxations. Compared with the existing method by Wainwright et al. (2005) to construct such a hierarchy by combining (hyper-)trees, this is much simpler.
- It is easy to tighten the relaxation *incrementally* during max-sum diffusion, resulting in fact in a (dual) cutting plane algorithm.
- It can be easily proved that the algorithm exactly solves problems with supermodular interactions of any arity.

The framework is described in (Werner, 2008a, 2010a). The hierarchy of LP relaxations and the cutting plane algorithm are revisited in the book chapter (Franc et al., 2012), where, I believe, they reached a mature state.

Publications included in the thesis: (Werner, 2010a) and (Franc et al., 2012)

3.3 Semiring-based Generalization

Expressions of type (2.4) and (2.3) can be unified using abstract operations $\oplus$ and $\otimes$. Suppose that the numbers $f_A(x_A)$ belong to some set S, the two operations $\oplus$ and $\otimes$ are associative and commutative on S, and $\otimes$ distributes over $\oplus$. In other words, the triplet $(S, \oplus, \otimes)$ defines a *commutative semiring*. By choosing different semirings, the expression

$$\bigoplus_{x_V} \bigotimes_{A \in E} f_A(x_A) \tag{3.1}$$

can now capture many different problems, for example:

- Semiring $(\mathbb{R}_+, +, \times)$ yields the sum-product problem (2.3).

- Semiring $(\mathbb{R}_+, \max, \times)$ yields the max-product problem (2.4).
- The max-sum semiring $(\mathbb{R} \cup \{-\infty\}, \max, +)$ is isomorphic (via logarithm) to the max-product semiring $(\mathbb{R}_+, \max, \times)$, hence yielding an isomorphic problem.
- The Boolean semiring $(\{0, 1\}, \max, \min)$ yields the classical *constraint satisfaction problem* (Mackworth, 1991).
- Semiring $([0, 1], \max, \min)$ yields the *fuzzy constraint satisfaction* problem (Rosenfeld et al., 1976).

The fact that certain problems and algorithms can be unified using the semiring formalism has been observed several times in pattern recognition (Aji and McEliece, 2000) and constraint programming, (Bistarelli et al., 1999) and (Rossi et al., 2006, chapter 9).

I observed that max-sum diffusion (its arbitrary-arity form) can be naturally generalized to a wide class of commutative semirings (Werner and Shekhovtsov, 2007; Werner, 2007a). Later I called the resulting algorithm *enforcing marginal consistency* of the network f (Werner, 2008b, 2013, 2014). It provides inspiring links between the disciplines of graphical models and constraint programming. The algorithm is so natural and simple that it is surprising that it was not proposed before. Let us describe its raw version:

- A *local equivalent transformation* of a pair (f_A, f_B) is any change of the functions f_A and f_B that preserves the function

$$f_A(x_A) \otimes f_B(x_B) \tag{3.2}$$

  for all $x_{A \cup B}$. Clearly, this also preserves the function $\bigotimes_{A \in E} f_A(x_A)$ for all x_V and therefore problem (3.1).

  For example, let $A = (1, 2)$, $B = (2, 3, 4)$, and $\otimes$ be the usual multiplications. Then a local equivalent transformation on (f_A, f_B) is any change of arrays f_{12} and f_{234} that preserves the expression $f_{12}(x_1, x_2) f_{234}(x_2, x_3, x_4)$ for all x_1, x_2, x_3, x_4.
- A pair (f_A, f_B) is *marginal-consistent* if

$$\bigoplus_{x_{A \setminus B}} f_A(x_A) = \bigoplus_{x_{B \setminus A}} f_B(x_B), \tag{3.3}$$

  that is, the variables shared by f_A and f_A have equal marginals.

  For example, let $A = (1, 2)$, $B = (2, 3, 4)$, and $\oplus$ be maximization. Then the pair (f_A, f_B) is marginal consistent if $f_{12}(x_1, x_2) = \max_{x_3} f_{234}(x_2, x_3, x_4)$.
- *Enforcing marginal consistency of a pair* (f_A, f_B) is the local equivalent transformation of the pair that makes it marginal-consistent. For most semirings of interest, this transformation is (surprisingly) unique and can be described by a simple rule.
- The algorithm to *enforce marginal consistency of network* f repeats the following iteration: pick a pair of hyperedges, (A, B) with $A, B \in E$, and enforce marginal consistency of (f_A, f_B). The pairs (A, B) can be picked arbitrarily (for example, in a predefined fixed order), provided that every pair is visited with a non-zero probability. The algorithm converges to a state in which every pair of interactions is marginal-consistent.

For different semirings, the algorithm has different behavior and its fixed points have different properties. For many semirings one obtains known algorithms. For example, the Boolean semiring $(\{0, 1\}, \max, \min)$ yields the *arc*

consistency algorithm, ubiquitous in constraints satisfaction. Semiring $(\mathbb{R} \cup \{-\infty\}, \max, +)$ yields max-sum diffusion. For semiring $(\mathbb{R}_+, +, \times)$, the algorithm can be used to strengthen arc consistency in the classical constraint satisfaction problem (Werner, 2011a).

Publications included in the thesis: (Werner, 2014)

3.4 Universality of the Local Marginal Polytope

The LP relaxation reviewed in (Werner, 2007b, 2010a) is exact for a large class of min-sum instances (e.g., all tractable languages with finite costs (Thapper and Živný, 2012) and instances with bounded treewidth) and it is a basis for constructing good approximations for many other instances (Kappes et al., 2013). It is therefore of great practical interest to have efficient algorithms to solve the LP relaxation.

As LP is in the P complexity class, one might think that solving the LP relaxation is easy, by off-the-self solvers. However, this is not so if we want to solve large or very large instances that often occur, e.g., in computer vision. For min-sum problems with pairwise interactions and 2 labels, the LP relaxation can be solved efficiently because it reduces in linear time to max-flow (Boros and Hammer, 2002; Rother et al., 2007). For more general problems, no really efficient algorithm is known. In particular, the well-known simplex and interior point methods are not applicable, if only due to their quadratic space complexity. Convergent message-passing algorithms (Kovalevsky and Koval, approx. 1975; Werner, 2007b; Kolmogorov, 2006) and the Augmenting DAG / VAC algorithm (Koval and Schlesinger, 1976; Cooper et al., 2010) do apply to large-scale instances but they find only a local (rather than global) optimum of the dual LP relaxation.

My colleague and me have shown (Průša and Werner, 2013, 2014; Živný et al., 2014) that the quest for efficient algorithms to solve the LP relaxation of the general min-sum problem has a fundamental limitation, because this task is not easier than solving any linear program. Precisely, our most important result says that every linear program can be reduced in linear time to the LP relaxation of a max-sum problem (allowing infinite costs) with 3 labels. From the polyhedral point of view, we have shown that every polytope is (up to scale) a coordinate-erasing projection of a face of a local marginal polytope (which is the feasible set of the LP relaxation) with 3 labels, whose description can be computed from the input polytope in linear time. Later we proved a similar (though somewhat weaker) result for a subclass of the max-sum problem, the attractive Potts problem (also known as the metric labeling problem) (Průša and Werner, 2015).

Publications included in the thesis: (Průša and Werner, 2014)

3.5 An Insight into Belief Propagation

Belief propagation (BP) is a well-known algorithm to approximate marginals of the Gibbs distribution (2.1) – in other words, it tackles the sum-product problem. It was proposed by Pearl (1988) and later generalized to interactions of any arity by Kschischang et al. (2001). For acyclic (hyper-)graphs, BP always converges and yields the exact marginals. For graphs with cycles, BP is not guaranteed to converge but it was empirically observed that when it converges,

it often yields surprisingly good approximations of the true marginals. Attempts to understand this behavior has generated a large body of literature, as surveyed e.g. in (Wainwright and Jordan, 2008).

Some solid ground to understand BP was provided by Yedidia et al. (2000, 2005), who discovered that BP fixed points coincide with stationary points of the *Bethe variational problem*, long known in statistical physics. This problem consists in minimizing a non-convex function over an affine subspace. Though this sheds light on the nature of BP *fixed points*, it does not explain the BP *algorithm* because the variables in this algorithm ('beliefs') are infeasible to the Bethe variational problem until convergence.

In (Werner, 2011b, 2010b), we offer an alternative, 'dual' view on the dynamics of the BP algorithm ('dual' is in quotes because the Bethe variational problem is non-convex, therefore the classical Fenchel or Lagrange duality does not apply). Here, the BP fixed points are shown to coincide with the stationary points of a certain multivariate function (which we called the *Bethe log-partition function*) without any constraints. The BP algorithm can be seen as a block-coordinate search to find such a stationary point: in every iteration, we find (in closed form) the stationary point of the Bethe log-partition function restricted on a small subset of variables, the remaining variables being fixed. On condition that the stationary points were local extremes (as is usual), this block-coordinate search would necessarily converge. However, we showed they are *saddle points*, in which case the search may oscillate.

Publications included in the thesis: (Werner, 2010b)

Bibliography

K.I. Aardal, van Hoesel. C.P.M., A.M.C.A. Koster, C. Mannino, and A. Sassano. Models and solution techniques for the frequency assignment problem. *Annals of Operations Research*, 153:79–129, 2007.

Srinivas M. Aji and Robert J. McEliece. The generalized distributive law. *IEEE Trans. on Information Theory*, 46(2):325–343, 2000.

Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer Science+Bussiness Media, New York, NY, 2006.

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.

Endre Boros and Peter L. Hammer. Pseudo-Boolean optimization. *Discrete Applied Mathematics*, 123(1-3):155–225, 2002. ISSN 0166-218X.

M. C. Cooper, S. de Givry, M. Sanchez, T. Schiex, M. Zytnicki, and T. Werner. Soft arc consistency revisited. *Artificial Intelligence*, 174(7-8):449–478, 2010.

Michel Marie Deza and Monique Laurent. *Geometry of Cuts and Metrics*. Springer, Berlin, 1997.

R. Durbin, S.R. Eddy, A. Krogh, and G. Mitchison. *Biological Sequence Analysis*. Cambridge University Press, 1999.

V. Franc, S. Sonnenburg, and T. Werner. Cutting plane methods in machine learning. In S. Sra, S. Nowozin, and S. J. Wright, editors, *Optimization for Machine Learning*. MIT Press, 2012.

S. Geman and D. Geman. Stochastic relaxation, Gibbs distributions and Bayesian restoration of images. *IEEE Trans. Pattern Analysis Machine Intelligence*, 6:721–741, 1984.

Jörg H. Kappes, Bjoern Andres, Fred A. Hamprecht, Christoph Schnörr, Sebastian Nowozin, Dhruv Batra, Sungwoong Kim, Bernhard X. Kausler, Jan Lellmann, Nikos Komodakis, and Carsten Rother. A comparative study of modern inference techniques for discrete energy minimization problem. In *Conf. Computer Vision and Pattern Recognition*, 2013.

Carleton L. Kingsford, Bernard Chazelle, and Mona Singh. Solving and analyzing side-chain positioning problems using linear and integer programming. *Bioinformatics*, 21(7):1028–1039, 2005.

D. Koller and N. Friedman. *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.

Vladimir Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 28 (10):1568–1583, 2006.

Vladimir Kolmogorov and Ramin Zabih. What energy functions can be minimized via graph cuts? In *Eur. Conf. on Computer Vision*, pages 65–81. Springer-Verlag, 2002. ISBN 3-540-43746-0.

Arie Koster, C. P. M. van Hoesel, and A. W. J. Kolen. The partial constraint satisfaction problem: Facets and lifting theorems. *Operations Research Letters*, 23(3-5):89–97, 1998.

V. K. Koval and M. I. Schlesinger. Dvumernoe programmirovanie v zadachakh analiza izobrazheniy (Two-dimensional programming in image analysis problems). *Automatics and Telemechanics*, 8:149–168, 1976. In Russian.

V. A. Kovalevsky and V. K. Koval. A diffusion algorithm for decreasing the energy of the max-sum labeling problem. Glushkov Institute of Cybernetics, Kiev, USSR. Unpublished, approx. 1975.

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

S. Lauritzen. *Graphical Models*. Oxford University Press, 1996.

A. Mackworth. Constraint satisfaction. In *Encyclopaedia of Artificial Intelligence*, pages 285–292. Wiley, 1991.

M. Mézard. Passing messages between disciplines. *Science*, 301:1685–1686, 2003.

M. Mézard and A. Montanari. *Information, Physics, and Computation*. Oxford University Press, Inc., New York, NY, USA, 2009.

Judea Pearl. *Probabilistic reasoning in intelligent systems: Networks of plausible inference*. Morgan Kaufmann, San Francisco, 1988.

Daniel Průša and Tomáš Werner. Universality of the local marginal polytope. In *Conf. on Computer Vision and Pattern Recognition*, pages 1738–1743. IEEE Computer Society, 2013.

Daniel Průša and Tomáš Werner. Universality of the local marginal polytope. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, PP(99): 1–1, 2014. ISSN 0162-8828. doi: 10.1109/TPAMI.2014.2353626. Electronic preprint.

Daniel Průša and Tomáš Werner. How hard is the LP relaxation of the Potts min-sum labeling problem? In *Conf. on Energy Minimization Methods in Computer Vision and Pattern Recognition, Hongkong*. Springer, 2015. To appear.

A. Rosenfeld, R. A. Hummel, and S. W. Zucker. Scene labeling by relaxation operations. *IEEE Trans. on Systems, Man, and Cybernetics*, 6(6):420–433, June 1976.

Francesca Rossi, Peter van Beek, and Toby Walsh, editors. *Handbook of Constraint Programming*. Elsevier, 2006.

Carsten Rother, Vladimir Kolmogorov, Victor S. Lempitsky, and Martin Szummer. Optimizing binary MRFs via extended roof duality. In *Conf. Computer Vision and Pattern Recognition, Minneapolis, USA*, 2007.

M. Sanchez, S. de Givry, and T. Schiex. Mendelian error detection in complex pedigrees using weighted constraint satisfaction techniques. *Constraints*, 13 (1):130–154, 2008.

D. Schlesinger and B. Flach. Transforming an arbitrary minsum problem into a binary one. Technical Report TUD-FI06-01, Dresden University of Technology, 2006.

Michail I. Schlesinger and Boris Flach. Some solvable subclasses of structural recognition problems. In *Czech Pattern Recognition Workshop*. Czech Pattern Recognition Society, 2000.

M. I. Shlezinger. Syntactic analysis of two-dimensional visual signals in noisy conditions. *Cybernetics and Systems Analysis*, 12(4):612–628, 1976. Translation from Russian.

Johan Thapper and Stanislav Živný. The power of linear programming for valued CSPs. In *Symp. Foundations of Computer Science*, pages 669–678. IEEE, 2012.

Stanislav Živný, Tomáš Werner, and Daniel Průša. The power of lp relaxation for map inference. In S. Nowozin, P. V. Gehler, J. Jancsary, and C. Lampert, editors, *Advanced Structured Prediction*. MIT Press, 2014. To appear.

M. Wainwright, T. Jaakkola, and A. Willsky. MAP estimation via agreement on (hyper)trees: message passing and linear programming approaches. *IEEE Trans. Information Theory*, 51(11):3697–3717, 2005.

Martin J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305, 2008.

Tomáš Werner. A linear programming approach to max-sum problem: A review. Technical Report CTU-CMP-2005-25, Center for Machine Perception, Czech Technical University, December 2005.

Tomáš Werner. What is decreased by the max-sum arc consistency algorithm? In *Intl. Conf. on Machine Learning, Oregon, USA*, June 2007a.

Tomáš Werner. A linear programming approach to max-sum problem: A review. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 29(7):1165–1179, July 2007b.

Tomáš Werner. High-arity interactions, polyhedral relaxations, and cutting plane algorithm for soft constraint optimisation (MAP-MRF). In *Computer Vision and Pattern Recognition Conf., Anchorage, USA*, June 2008a.

Tomáš Werner. Marginal consistency: Unifying constraint propagation on commutative semirings. In *Intl. Workshop on Preferences and Soft Constraints*, pages 43–57, 2008b.

Tomáš Werner. Revisiting the decomposition approach to inference in exponential families and graphical models. Research Report CTU-CMP-2009-06, Center for Machine Perception, K13133 FEE Czech Technical University, Prague, Czech Republic, May 2009.

Tomáš Werner. Revisiting the linear programming relaxation approach to Gibbs energy minimization and weighted constraint satisfaction. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 32(8):1474–1488, August 2010a.

Tomáš Werner. Primal view on belief propagation. In *Conf. on Uncertainty in Artificial Intelligence*, pages 651–657, July 2010b.

Tomáš Werner. How to compute primal solution from dual one in MAP inference in MRF? *Control Systems and Computers*, March–April(2):35–45, 2011a. ISSN 0130-5395.

Tomáš Werner. Zero-temperature limit of a convergent algorithm to minimize the bethe free energy. Research Report CTU-CMP-2011-14, Center for Machine Perception, K13133 FEE Czech Technical University, Prague, Czech Republic, December 2011b.

Tomáš Werner. Marginal consistency: Unifying convergent message passing and constraint propagation. Research Report CTU-CMP-2013-26, Department of Cybernetics, FEE Czech Technical University, Prague, Czech Republic, October 2013.

Tomáš Werner. Marginal consistency: Upper-bounding partition functions over commutative semirings. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, PP(99):1–1, 2014. ISSN 0162-8828. doi: 10.1109/TPAMI.2014.2363833. Electronic preprint.

Tomáš Werner and Alexander Shekhovtsov. Unified framework for semiring-based arc consistency and relaxation labeling. In *12th Computer Vision Winter Workshop, St. Lambrecht, Austria*, pages 27–34. Graz University of Technology, February 2007.

J. S. Yedidia, W. T. Freeman, and Y. Weiss. Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Trans. Information Theory*, 51(7):2282–2312, 2005.

Jonathan Yedidia, William T. Freeman, and Yair Weiss. Generalized belief propagation. In *Neural Information Processing Systems*, pages 689–695, 2000.

Attached Publications

As the rest of the thesis, we include copies of the following publications (in this order):

- (Werner, 2007b),
- (Werner, 2010a),
- (Franc et al., 2012),
- (Průša and Werner, 2014),
- (Werner, 2014),
- (Werner, 2010b).

A Linear Programming Approach to Max-sum Problem: A Review

Tomáš Werner
Dept. of Cybernetics, Czech Technical University
Karlovo náměstí 13, 121 35 Prague, Czech Republic

***Abstract*— The max-sum labeling problem, defined as maximizing a sum of binary functions of discrete variables, is a general NP-hard optimization problem with many applications, such as computing the MAP configuration of a Markov random field. We review a not widely known approach to the problem, developed by Ukrainian researchers Schlesinger et al. in 1976, and show how it contributes to recent results, most importantly those on convex combination of trees and tree-reweighted max-product. In particular, we review Schlesinger's upper bound on the max-sum criterion, its minimization by equivalent transformations, its relation to constraint satisfaction problem, that this minimization is dual to a linear programming relaxation of the original problem, and three kinds of consistency necessary for optimality of the upper bound. We revisit problems with Boolean variables and supermodular problems. We describe two algorithms for decreasing the upper bound. We present an example application to structural image analysis.**

***Index Terms*— Markov random fields, undirected graphical models, constraint satisfaction problem, belief propagation, linear programming relaxation, max-sum, max-plus, max-product, supermodular optimization.**

I. INTRODUCTION

The binary (i.e., pairwise) max-sum labeling problem is defined as maximizing a sum of unary and binary functions of discrete variables, i.e., as computing

$$\max_{\mathbf{x}\in X^T} \Big[\sum_{t\in T} g_t(x_t) + \sum_{\{t,t'\}\in E} g_{tt'}(x_t, x_{t'}) \Big],$$

where an undirected graph (T, E), a finite set X, and numbers $g_t(x_t)$, $g_{tt'}(x_t, x_{t'}) \in \mathbb{R}\cup\{-\infty\}$ are given. It is a very general NP-hard optimization problem, which has been studied and applied in several disciplines, such as statistical physics, combinatorial optimization, artificial intelligence, pattern recognition, and computer vision. In the latter two, the problem is also known as computing maximum posterior (MAP) configuration of Markov random fields (MRF).

This article reviews an old and not widely known approach to the max-sum problem by Ukrainian scientists Schlesinger *et al*. and shows how it contributes to recent knowledge.

A. Approach by Schlesinger et al.

The basic elements of the old approach were given by Schlesinger in 1976 in structural pattern recognition. In [1], he generalizes locally conjunctive predicates by Minsky and Papert [2] to *two-dimensional (2D) grammars* and shows these are useful for structural image analysis. Two tasks are considered on 2D grammars. The first task assumes analysis of ideal, noise-free images: test whether an input image belongs to the language generated by a given grammar. It leads to what is today known as the *Constraint Satisfaction Problem* (CSP) [3], or *discrete relaxation labeling*. Finding the largest *arc consistent* subproblem provides some necessary but not sufficient conditions for satisfiability and unsatisfiability of the problem. The second task considers analysis of noisy images: find an image belonging to the language generated by a given 2D grammar that is 'nearest' to a given image. It leads to the max-sum problem.

In detail, paper [1] formulates a linear programming relaxation of the max-sum problem and its dual program. The dual is interpreted as minimizing an upper bound to the max-sum problem by *equivalent transformations*, which are redefinitions of the the problem that leave the objective function unchanged. The optimality of the upper bound is equal to *triviality* of the problem. Testing for triviality leads to a CSP.

An algorithm to decrease the upper bound, which we called *augmenting DAG algorithm*, was suggested in [1] and presented in more detail by Koval and Schlesinger [4] and further in [5]. Another algorithm to decrease the upper bound is a coordinate descent method, *max-sum diffusion*, discovered by Kovalevsky and Koval [6] and later independently by Flach [7]. Schlesinger noticed [8] that the termination criterion of both algorithms, arc consistency, is necessary but not sufficient for minimality of the upper bound. Thus, the algorithms sometimes find the true minimum of the upper bound and sometimes only decrease it to some point.

The material in [1], [4] is presented in detail in the book [9]. The name '2D grammars' was later assigned a different meaning in the book [10] by Schlesinger and Hlaváč. In their original meaning, they largely coincide with MRFs.

By minimizing the upper bound, some max-sum problems can be solved to optimality (the upper bound is tight) and some cannot (there is an integrality gap). Schlesinger and Flach [11] proved that supermodular problems have zero integrality gap.

B. Relation to Recent Works

Independently on the work by Schlesinger *et al*., a significant progress has recently been achieved in the max-sum problem. This section reviews the most relevant newer results by others and shows how they relate to the old approach.

1) Convex relaxations and upper bounds: It is common in combinatorial optimization to approach NP-hard problems via continuous relaxations of their integer programming formulations. The linear programming relaxation given by Schlesinger [1] is quite natural and has been suggested independently and later by others: by Koster *et al*. [12], [13], who address the max-sum problem as a generalization of CSP, the *Partial CSP*; by Chekuri *et al*. [14] and Wainwright *et al*. [15]; and in bioinformatics [16]. Koster *et al*. in addition give two classes of non-trivial facets of the Partial CSP polytope, i.e., linear constraints missing in the relaxation.

Max-sum problems with Boolean (i.e., two-state) variables are a subclass of pseudo-Boolean and quadratic Boolean optimization, see e.g. the review [17]. Here, several different upper bounds were suggested, which were shown equivalent by Hammer *et al*. [18]. These bounds are in turn equivalent to [1], [12], [14] with Boolean variables, as shown in [19].

Other than linear programming relaxations of the max-sum problem have been suggested, such as quadratic [20], [21] or semidefinite [22] programming relaxations. We will not discuss these.

2) Convex combination of trees: The max-sum problem has been studied in terminology of graphical models, in particular it is equivalent to finding MAP configurations of undirected graphical models, also known as MRFs. This research primarily focused on computing the partition function and marginals of MRFs and approached the max-sum problem as the limit case of this task.

Wainwright *et al*. [23] shows that a convex combination of problems provides a convex upper bound on the log-partition function of MRF. These subproblems can be conveniently chosen as (tractable) tree problems. For the sum-product problem on cyclic graphs, this upper bound is almost never tight. In the max-sum limit (also known as the *zero temperature limit*) the bound is tight much more often, namely if the optima on individual trees share a common configuration, which is referred to as *tree agreement* [15], [24]. Moreover, in the max-sum case the bound is independent on the choice of trees. Minimizing the upper bound is shown [15], [24] to be a Lagrangian dual of a linear programming relaxation of the max-sum problem. This relaxation is the same as in [1], [12], [14]. Besides directly solving this relaxation, *tree-reweighted message passing* (TRW) algorithm is suggested to minimize the upper bound. Importantly, it is noted [25], [26] that message passing can be alternatively viewed as *reparameterizations* of the problem. TRW is guaranteed neither to converge nor to decrease the upper bound monotonically. Kolmogorov [27]–[30] suggests its sequential modification (TRW-S) and conjectures that it always converges to a state characterized by *weak tree agreement*. He further shows that the point of convergence might differ from a global minimum of the upper bound, however, for Boolean variables [19], [31] they are equal.

The approach based on convex combination of trees is closest to the approach by Schlesinger's *et al*. The linear programming relaxation considered by Wainwright is the same as Schlesinger's one. Reparameterizations correspond to Schlesinger's equivalent transformations. If the trees are chosen as individual nodes and edges, Wainwright's upper bound becomes Schlesinger's upper bound, tree agreement becomes CSP satisfiability, and weak tree agreement becomes arc consistency. The convenient choice of subproblems as nodes and edges is without loss of generality because Wainwright's bound is independent on the choice of trees.

The approach based on convex combination of trees is more general than the approach reviewed in this article but the latter is simpler, hence it may be more suitable for analysis. However, the translation between the two is not straightforward and the approach by Schlesinger *et al*. provides the following contributions to that by Wainwright *et al*. and Kolmogorov.

Duality of linear programming relaxation of the max-sum problem and minimizing Schlesinger's upper bound is proved more straightforwardly by putting both problems into matrix form [1], as is common in linear programming.

The max-sum problem is intimately related with CSP via complementary slackness. This reveals that testing for tightness of the upper bound is NP-complete, which has not been noticed by others. It leads to a relaxation of CSP, which provides a simple way [8] to characterize spurious minima of the upper bound. This has an independent value for CSP research.

The max-sum diffusion is related to TRW-S but has advantage in its simplicity, which also might help further analysis. With its combinatorial flavor, the Koval-Schlesinger augmenting DAG algorithm [4] is dissimilar to any recent algorithm and somewhat resembles the augmenting path algorithm for max-flow/min-cut problem.

3) Loopy belief propagation: It is long known that the sum-product and max-sum problems on trees can be efficiently solved by *belief propagation* and *message passing* [32]. When applied to cyclic graphs, these algorithms were empirically found sometimes to converge and sometimes not to, with the fixed points (if any) sometimes being useful approximations. The main recent result [33] is that the fixed points of this 'loopy' belief propagation are local minima a non-convex function, known in statistical physics as Bethe free energy.

The max-sum diffusion resembles loopy belief propagation: both repeat simple local operations and both can be interpreted as a coordinate descent minimization of some functional. However, for the diffusion this functional is convex while for belief propagation it is non-convex.

4) CSP and extensions: The CSP seeks to find values of discrete variables that satisfy given logical constraints. Its extensions have been suggested in which the constraints become soft and one seeks to maximize a criterion rather than satisfy constraints. The max-sum problem is often closely related to these extensions. Examples are the Max CSP [34] (subclass of the max-sum problem), Valued CSP [35] (more general than max-sum), and Partial CSP [12] (equivalent to max-sum).

The max-sum problem relates to CSP also via complementary slackness, as mentioned above. This establishes links to the large CSP literature, which may be fruitful in both directions. This article seems to be the first in pattern recognition and computer vision to make this link.

5) Maximum flow (minimum cut): Finding max-flow/min-cut in a graph has been recognized very useful for (mainly low-level) computer vision [36]. Later it has been realized that

supermodular max-sum problems can be translated to max-flow/min-cut (see section VIII). For supermodular max-sum problems, Schlesinger's upper bound is tight and finding an optimal configuration is tractable [11]. The relation of this result with lattice theory is considered in [19], [37]–[40]. We further extend this relation and give it a simpler form.

C. Organization of the Article

Section II introduces the labeling problem on commutative semirings and basic concepts. Section III reviews CSP. Section IV presents the linear programming relaxation of the max-sum problem, its dual, Schlesinger's upper bound, and equivalent and trivial problems. Section V characterizes minimality of the upper bound. Two algorithms for decreasing the upper bound are described in sections VI and VII. Section VIII establishes that the bound is tight for supermodular problems. Application to structural image analysis [1], [9] is presented in section IX. A previous version of this article is [41].

Logical conjunction (disjunction) is denoted by $\wedge$ ($\vee$). Function $[\![\psi]\!]$ returns 1 if logical expression ψ is true and 0 if it is false. The set of *all* maximizers of $f(x)$ is $\operatorname{argmax}_x f(x)$. Assignment is denoted by $x := y$, symbol $x \mathrel{+}= y$ denotes $x := x + y$. New concepts are in **boldface**.

II. Labeling Problems on Commutative Semirings

This section defines a class of labeling problems of which the CSP and the max-sum problem are special cases. Here we introduce basic terminology used in the rest of the article.

We will use the terminology from [11] where the variables are called objects and their values are called labels. Let $G = (T, E)$ be an undirected graph, where T is a discrete set of **objects** and $E \subseteq \binom{T}{2}$ is a set of (object) **pairs**. The set of neighbors of an object t is $N_t = \{\, t' \mid \{t, t'\} \in E \,\}$. Each object $t \in T$ is assigned a **label** $x_t \in X$, where X is a discrete set. A **labeling** $\mathbf{x} \in X^T$ is a $|T|$-tuple that assigns a single label x_t to each object t. When not viewed as components of $\mathbf{x}$, elements of X will be denoted by x, x' without subscripts.

Let $(T \times X, E_X)$ be another undirected graph with edges $E_X = \{\, \{(t, x), (t', x')\} \mid \{t, t'\} \in E,\ x, x' \in X \,\}$. When G is a chain, this graph corresponds to the *trellis diagram*, frequently used to visualize Markov chains. The nodes and edges of G will be called objects and pairs, respectively, whereas the terms **nodes** and **edges** will refer to $(T \times X, E_X)$. The set of all nodes and edges is $I = (T \times X) \cup E_X$. The set of edges leading from a node (t, x) to all nodes of a neighboring object $t' \in N_t$ is a **pencil** (t, t', x). The set of all pencils is $P = \{\, (t, t', x) \mid \{t, t'\} \in E,\ x \in X \,\}$. Figure 1 shows how both graphs, their parts, and labelings will be visualized.

Let an element $g_t(x)$ of a set S be assigned to each node (t, x) and an element $g_{tt'}(x, x')$ to each edge $\{(t, x), (t', x')\}$, where $g_{tt'}(x, x') = g_{t't}(x', x)$. The vector obtained by concatenating all $g_t(x)$ and $g_{tt'}(x, x')$ is denoted by $\mathbf{g} \in S^I$.

Before starting with the max-sum labeling problem, we introduce labeling problems in a more general form. It was observed [11], [42]–[45] that different labeling problems can be unified, by letting a suitable commutative semiring specify

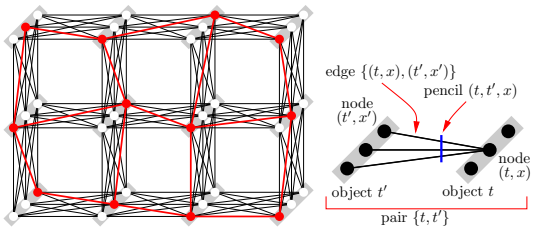


Fig. 1. (a) The 3×4 grid graph G (i.e., $|T| = 12$), graph $(T \times X, E_X)$ for $|X| = 3$ labels, and a labeling $\mathbf{x}$ (emphasized). (b) Parts of both graphs.

how different constraints are combined together. Let S endowed with two binary operations $\oplus$ and $\otimes$ form a commutative semiring $(S, \oplus, \otimes)$. The semiring formulation of the labeling problem [11] is defined as computing

$$\bigoplus_{\mathbf{x} \in X^T} \Big[\bigotimes_{t \in T} g_t(x_t) \otimes \bigotimes_{\{t,t'\} \in E} g_{tt'}(x_t, x'_t) \Big]. \tag{1}$$

More exactly, this is the *binary* labeling problem, according to the highest arity of the functions in the brackets. We will not consider problems of higher arity.

Interesting problems are obtained, modulo isomorphisms, by the following choices of the semiring:

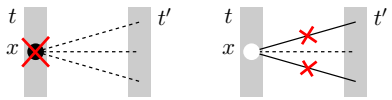
$(S, \oplus, \otimes)$	task
$(\{0,1\}, \vee, \wedge)$	or-and problem, CSP
$([-\infty, \infty), \min, \max)$	min-max problem
$([-\infty, \infty), \max, +)$	max-sum problem
$([0, \infty), +, *)$	sum-product problem

Note that the *extended domain*, $S = [-\infty, \infty)$, of min-max and max-sum problems yields a more general formulation than usually used $S = (-\infty, \infty)$.

The topic of this article is the max-sum problem but we will briefly cover also the closely related CSP. Since semiring $(\{0,1\}, \vee, \wedge)$ is isomorphic with $(\{-\infty, 0\}, \max, +)$, CSP is a subclass of the max-sum problem. However, we will treat CSP separately since a lot of independent research has been done on it. We will not discuss the sum-product problem (i.e., computing MRF partition function) and the min-max problem.

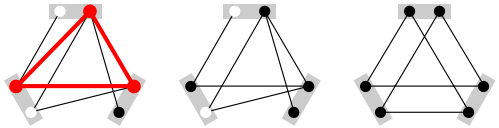
III. Constraint Satisfaction Problem

The **constraint satisfaction problem** (CSP) [3] is defined as finding a labeling that satisfies given unary and binary constraints, i.e., that passes through some or all of given nodes and edges. It was introduced, often independently, several times in computer vision [1], [46]–[48] and artificial intelligence [49], often under different names, such as the *Consistent Labeling Problem* [50]. CSP is NP-complete. Tractable subclasses are obtained either by restricting the structure of G (limiting its fractional treewidth [51]) or the constraint language. In the latter, a lot of research has been done and mathematicians seem to be close to complete classification [52]. Independently on this, Schlesinger and Flach discovered a tractable CSP subclass defined by the *interval condition* [11], [38]. In particular, binary CSP with Boolean variables is known to be tractable.



(a) (b)

Fig. 2. The arc consistency algorithm deletes (a) nodes not linked with some neighbor by any edge, and (b) edges lacking an end node.



(a) (b) (c)

Fig. 3. Examples of CSPs: (a) satisfiable, hence with a non-empty kernel which allows to form a labeling (the labeling is emphasized); (b) with an empty kernel, hence unsatisfiable; (c) arc consistent but unsatisfiable. The forbidden nodes are in white, the forbidden edges are not shown.

We denote a CSP instance by $(G, X, \bar{\mathbf{g}})$. Indicators $\bar{g}_t(x)$, $\bar{g}_{tt'}(x, x') \in \{0, 1\}$ say if the corresponding node or edge is allowed or forbidden. The task is to compute the set

$$\bar{L}_{G,X}(\bar{\mathbf{g}}) = \Big\{ \mathbf{x} \in X^T \;\Big|\; \bigwedge_t \bar{g}_t(x_t) \wedge \bigwedge_{\{t,t'\}} \bar{g}_{tt'}(x_t, x_{t'}) = 1 \Big\}. \quad (2)$$

A CSP is **satisfiable** if $\bar{L}_{G,X}(\bar{\mathbf{g}}) \neq \emptyset$.

Some conditions necessary or sufficient (but not both) for satisfiability can be given in terms of local consistencies, surveyed e.g. in [53]. The simplest local consistency is arc consistency. A CSP is **arc consistent** if

$$\bigvee_{x'} \bar{g}_{tt'}(x, x') = \bar{g}_t(x), \quad \{t, t'\} \in E,\ x \in X. \quad (3)$$

CSP $(G, X, \bar{\mathbf{g}}')$ is a **subproblem** of $(G, X, \bar{\mathbf{g}})$ if $\bar{\mathbf{g}}' \leq \bar{\mathbf{g}}$. The **union** of CSPs $(G, X, \bar{\mathbf{g}})$ and $(G, X, \bar{\mathbf{g}}')$ is $(G, X, \bar{\mathbf{g}} \vee \bar{\mathbf{g}}')$. Here, operations $\leq$ and $\vee$ are meant componentwise. Following [1], [9], we define the **kernel** of a CSP as follows. First note that the union of arc consistent CSPs is arc consistent. To see this, write the disjunction of (3) for arc consistent $\bar{\mathbf{g}}$ and $\bar{\mathbf{g}}'$ as $[\bigvee_{x'} \bar{g}_{tt'}(x, x')] \vee [\bigvee_{x'} \bar{g}'_{tt'}(x, x')] = \bigvee_{x'} [\, \bar{g}_{tt'}(x, x') \vee \bar{g}'_{tt'}(x, x') \,] = \bar{g}_t(x) \vee \bar{g}'_t(x)$, obtaining that $\bar{\mathbf{g}} \vee \bar{\mathbf{g}}'$ satisfies (3). The kernel of a CSP is the union of all its arc consistent subproblems. Arc consistent subproblems of a problem form a join semilattice w.r.t. the partial ordering by inclusion $\leq$. The greatest element of this semilattice is the kernel. Equivalently, the kernel is the largest arc consistent subproblem.

The kernel can be found by the **arc consistency algorithm**, known also as *discrete relaxation labeling* [48]. Starting with their initial values, the variables $\bar{g}_t(x)$ and $\bar{g}_{tt'}(x, x')$ violating (3) are iteratively set to zero by applying rules (figure 2)

$$\bar{g}_t(x) := \bar{g}_t(x) \wedge \bigvee_{x'} \bar{g}_{tt'}(x, x'), \quad (4a)$$

$$\bar{g}_{tt'}(x, x') := \bar{g}_{tt'}(x, x') \wedge \bar{g}_t(x) \wedge \bar{g}_{t'}(x'). \quad (4b)$$

The algorithm halts when no further variable can be set to zero. It is well-known that the result does not depend on the order of the operations.

Theorem 1: Let $(G, X, \bar{\mathbf{g}}^*)$ be the kernel of a CSP $(G, X, \bar{\mathbf{g}})$. It holds that $\bar{L}_{G,X}(\bar{\mathbf{g}}) = \bar{L}_{G,X}(\bar{\mathbf{g}}^*)$.

Proof. The theorem is a corollary of the more general theorem 6, given later.

It can also be proved by the following induction argument. If a pencil (t, t', x) contains no edge, the node (t, x) clearly cannot belong to any labeling (figure 2a). Therefore, the node (t, x) can be deleted without changing $\bar{L}_{G,X}(\bar{\mathbf{g}})$. Similarly, if a node (t, x) is forbidden then no labeling can pass through any of the pencils $\{\, (t, t', x) \mid t' \in N_t \,\}$ (figure 2b). ■

A corollary of theorem 1 are the following conditions proving or disproving satisfiability. Figure 3 shows examples.

Theorem 2: Let $(G, X, \bar{\mathbf{g}}^*)$ denote the kernel of CSP $(G, X, \bar{\mathbf{g}})$.

- If the kernel is empty ($\bar{\mathbf{g}}^* = \mathbf{0}$) then the CSP is not satisfiable.
- If there is a unique label in each object ($\sum_x \bar{g}^*_t(x) = 1$ for $t \in T$) then the CSP is satisfiable.

IV. Max-sum Problem

We now turn our attention to the central topic of the article, **the max-sum problem**. Its instance is denoted by $(G, X, \mathbf{g})$, where $g_t(x)$ and $g_{tt'}(x, x')$ will be called **qualities**. The **quality of a labeling** $\mathbf{x}$ is

$$F(\mathbf{x} \,|\, \mathbf{g}) = \sum_{t \in T} g_t(x_t) + \sum_{\{t,t'\} \in E} g_{tt'}(x_t, x_{t'}). \quad (5)$$

Solving the problem means finding (one, several or all elements of) the set of optimal labelings

$$L_{G,X}(\mathbf{g}) = \operatorname*{argmax}_{\mathbf{x} \in X^T} F(\mathbf{x} \,|\, \mathbf{g}). \quad (6)$$

A. Linear Programming Relaxation

Let us formulate a linear programming relaxation of the max-sum problem (6). For that, we introduce a different representation of labelings that allows to represent 'partially decided' labelings. A **relaxed labeling** is a vector $\boldsymbol{\alpha}$ with the components $\alpha_t(x)$ and $\alpha_{tt'}(x, x')$ satisfying

$$\sum_{x'} \alpha_{tt'}(x, x') = \alpha_t(x), \quad \{t, t'\} \in E,\ x \in X \quad (7a)$$

$$\sum_x \alpha_t(x) = 1, \qquad t \in T \quad (7b)$$

$$\boldsymbol{\alpha} \geq \mathbf{0} \quad (7c)$$

where $\alpha_{tt'}(x, x') = \alpha_{t't}(x', x)$. Number $\alpha_t(x)$ is assigned to node (t, x), number $\alpha_{tt'}(x, x')$ to edge $\{(t, x), (t', x')\}$. The set of all $\boldsymbol{\alpha}$ satisfying (7) is a polytope, denoted by $\Lambda_{G,X}$. A binary vector $\boldsymbol{\alpha}$ represents a 'decided' labeling; there is a bijection between the sets X^T and $\Lambda_{G,X} \cap \{0, 1\}^I$, given by $\alpha_t(x) = [\![x_t = x]\!]$ and $\alpha_{tt'}(x, x') = \alpha_t(x)\, \alpha_{t'}(x')$. A non-integer $\boldsymbol{\alpha}$ represents an 'undecided' labeling.

Remark 1: Constraints (7a)+(7b) are linearly dependent. To see this, denote $\alpha_t = \sum_x \alpha_x(t)$ and $\alpha_{tt'} = \sum_{x,x'} \alpha_{tt'}(x, x')$ and sum (7a) over x, which gives $\alpha_t = \alpha_{tt'}$. Since G is connected, (7a) alone implies that α_t and $\alpha_{tt'}$ are equal for the

whole G. Thus, $\Lambda_{G,X}$ could be represented in a less redundant way by, e.g., replacing (7b) with $\sum_t \alpha_t = |T|$. It is shown in [12] that $\dim \Lambda_{G,X} = |T|(|X|-1) + |E|(|X|-1)^2$.

Remark 2: Conditions (7a)+(7c) can be viewed as a continuous generalization of arc consistency (3) in the following sense: for any $\boldsymbol{\alpha}$ satisfying (7a)+(7c), the CSP $\bar{\mathbf{g}}$ given by $\bar{g}_t(x) = [\![\alpha_t(x) > 0]\!]$ and $\bar{g}_{tt'}(x,x') = [\![\alpha_{tt'}(x,x') > 0]\!]$ satisfies (3).

Quality and equivalence of max-sum problems can be extended from ordinary to relaxed labelings. The quality of a relaxed labeling $\boldsymbol{\alpha}$ is the scalar product $\langle \mathbf{g}, \boldsymbol{\alpha} \rangle$. Like $F(\bullet \,|\, \mathbf{g})$, function $\langle \mathbf{g}, \bullet \rangle$ is invariant to equivalent transformations because $\langle \mathbf{0}^{\boldsymbol{\varphi}}, \boldsymbol{\alpha} \rangle$ identically vanishes, as is verified by substituting (9) and (7a). The **relaxed max-sum problem** is the linear program

$$\Lambda_{G,X}(\mathbf{g}) = \operatorname*{argmax}_{\alpha \in \Lambda_{G,X}} \langle \mathbf{g}, \boldsymbol{\alpha} \rangle. \tag{8}$$

The set $\Lambda_{G,X}(\mathbf{g})$ is a polytope, being the convex hull of the optimal vertices of $\Lambda_{G,X}$. If $\Lambda_{G,X}(\mathbf{g})$ has integer elements, they coincide with $L_{G,X}(\mathbf{g})$.

The linear programming relaxation (8) was suggested by several researchers independently: by Schlesinger in structural pattern recognition [1], by Koster *et al*. as an extension of CSP [12], by Chekuri *et al*. [14] for metric Markov random fields, and in bioinformatics [16].

Solving (8) by a general linear programming algorithm, such as simplex or interior point method, would be inefficient and virtually impossible for large instances which occur e.g. in computer vision. There are two ways to do better. First, the linear programming dual of (8) is more suitable for optimization because it has less variables. Second, a special algorithm utilizing the structure of the task has to be designed.

Further in section IV, we formulate the dual of (11) and interpret it as minimizing an *upper bound* on problem quality by *equivalent transformations*, and that tightness of the relaxation is equivalent to satisfiability of a CSP. The subsequent section V gives conditions for minimality of the upper bound, implied by complementary slackness.

B. Equivalent Max-sum Problems

Problems $(G, X, \mathbf{g})$ and $(G, X, \mathbf{g}')$ are called **equivalent** (denoted by $\mathbf{g} \sim \mathbf{g}'$) if functions $F(\bullet \,|\, \mathbf{g})$ and $F(\bullet \,|\, \mathbf{g}')$ are identical [1], [25], [28]. An **equivalent transformation** is a change of $\mathbf{g}$ taking a max-sum problem to its equivalent. Figure 4 shows the simplest such transformation: choose a pencil (t, t', x), add a number $\varphi_{tt'}(x)$ to $g_t(x)$, and subtract the same number from all edges in pencil (t, t', x).

A special equivalence class is formed by **zero problems** for which $F(\bullet \,|\, \mathbf{g})$ is the zero function. By (5), the zero class $\{\, \mathbf{g} \mid \mathbf{g} \sim \mathbf{0} \,\}$ is a linear subspace of $\mathbb{R}^I$. Problems $\mathbf{g}$ and $\mathbf{g}'$ are equivalent if and only if $\mathbf{g} - \mathbf{g}'$ is a zero problem.

We will parameterize any equivalence class by a vector $\boldsymbol{\varphi} \in \mathbb{R}^P$ with components $\varphi_{tt'}(x)$, assigned to pencils (t, t', x). Variables $\varphi_{tt'}(x)$ are called *potentials* in [1], [4], [9] and correspond to *messages* in belief propagation literature. The equivalent of a problem $\mathbf{g}$ given by $\boldsymbol{\varphi}$ is denoted by $\mathbf{g}^{\boldsymbol{\varphi}} = \mathbf{g} + \mathbf{0}^{\boldsymbol{\varphi}}$.

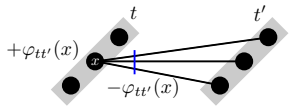


Fig. 4. The elementary equivalent transformation.

It is obtained by composing the elementary transformations shown in figure 4 for all pencils, which yields

$$g_t^{\boldsymbol{\varphi}}(x) = g_t(x) + \sum_{t' \in N_t} \varphi_{tt'}(x), \tag{9a}$$

$$g_{tt'}^{\boldsymbol{\varphi}}(x,x') = g_{tt'}(x,x') - \varphi_{tt'}(x) - \varphi_{t't}(x'). \tag{9b}$$

It is easy to see that problems $\mathbf{g}$ and $\mathbf{g}^{\boldsymbol{\varphi}}$ are equivalent for any $\boldsymbol{\varphi}$ since inserting (9) to (5) shows that $F(\mathbf{x} \,|\, \mathbf{g}^{\boldsymbol{\varphi}})$ identically equals $F(\mathbf{x} \,|\, \mathbf{g})$. We would like also the converse to hold, i.e. any two equivalent problems to be related by (9) for some $\boldsymbol{\varphi}$. However, this holds only if G is connected and all qualities $\mathbf{g}$ are finite, as is given by theorem 3. Connectedness of G is naturally satisfied in applications. The second assumption does not seem to be an obstacle in algorithms even when the extended domain $\mathbf{g} \in [-\infty, \infty)^I$ is used, though we still do not fully understand why.

Theorem 3: [54], [1], [28], [30] Let the graph G be connected and $\mathbf{g} \in \mathbb{R}^I$. $F(\bullet \,|\, \mathbf{g})$ is the zero function if and only if there exist numbers $\varphi_{tt'}(x) \in \mathbb{R}$ such that

$$g_t(x) = \sum_{t' \in N_t} \varphi_{tt'}(x), \tag{10a}$$

$$g_{tt'}(x,x') = -\varphi_{tt'}(x) - \varphi_{t't}(x'). \tag{10b}$$

The reader may skip the proof in the first reading.

Proof. The *if* part is easy, by verifying that (5) identically vanishes after substituting (10). We will prove the *only if* part.

Since $F(\bullet \,|\, \mathbf{g})$ is the zero function and therefore it is modular (i.e., both sub- and supermodular w.r.t. to any order $\leq$). By theorem 12 given later, also functions $g_{tt'}(\bullet, \bullet)$ are modular. Any modular function is a sum of univariate functions [55]. This implies (10b).

Let $\mathbf{x}$ and $\mathbf{y}$ be two labelings that differ only in an object t where they satisfy $x_t = x$ and $y_t = y$. After substituting (5) and (10b) to the equality $F(\mathbf{x} \,|\, \mathbf{g}) = F(\mathbf{y} \,|\, \mathbf{g})$, most terms cancel out giving $g_t(x) - \sum_{t'} \varphi_{tt'}(x) = g_t(y) - \sum_{t'} \varphi_{tt'}(y)$. Since this holds for any x and y, neither side depends on x. Thus we can denote $\varphi_t = g_t(x) - \sum_{t'} \varphi_{tt'}(x)$. Substituting (10) to $F(\bullet \,|\, \mathbf{g}) = 0$ yields $\sum_t \varphi_t = 0$.

To show (10a), we will give an equivalent transformation that sets all φ_t to zero. Let G' be a spanning tree of G. It exists because G is connected. Find a pair $\{t, t'\}$ in G' such that t is a leaf. Do the following transformation of $(G, X, \mathbf{g})$: set $\varphi_{tt'}(x)$ += φ_t for all x and $\varphi_{t't}(x')$ −= φ_t for all x'. Set $\varphi_{t'}$ += φ_t and φ_t := 0. Remove t and $\{t, t'\}$ from G'. Repeat until G' is empty. ∎

As a counter-example for infinite $\mathbf{g}$, consider the problem in figure 5a and the same problem with the crossed edge being $-\infty$. These two problems are equivalent but they are not related by (9) for any $\boldsymbol{\varphi} \in \mathbb{R}^P$.

$$\langle \mathbf{g}, \boldsymbol{\alpha} \rangle \to \max_{\boldsymbol{\alpha}} \qquad \sum_{t \in T} u_t + \sum_{\{t,t'\} \in E} u_{tt'} \to \min_{\boldsymbol{\varphi}, \mathbf{u}} \tag{11a}$$

$$\sum_{x' \in X} \alpha_{tt'}(x, x') = \alpha_t(x) \qquad \varphi_{tt'}(x) \in \mathbb{R}, \qquad \{t, t'\} \in E,\ x \in X \tag{11b}$$

$$\sum_{x \in X} \alpha_t(x) = 1 \qquad u_t \in \mathbb{R}, \qquad t \in T \tag{11c}$$

$$\sum_{x, x' \in X} \alpha_{tt'}(x, x') = 1 \qquad u_{tt'} \in \mathbb{R}, \qquad \{t, t'\} \in E \tag{11d}$$

$$\alpha_t(x) \geq 0 \qquad u_t - \sum_{t' \in N_t} \varphi_{tt'}(x) \geq g_t(x), \qquad t \in T,\ x \in X \tag{11e}$$

$$\alpha_{tt'}(x, x') \geq 0 \qquad u_{tt'} + \varphi_{tt'}(x) + \varphi_{t't}(x') \geq g_{tt'}(x, x'), \quad \{t, t'\} \in E,\ x, x' \in X \tag{11f}$$

TABLE I

C. Schlesinger's Upper Bound and Its Minimization

Let the **height of object** t and the **height of pair** $\{t, t'\}$ be respectively

$$u_t = \max_x g_t(x), \quad u_{tt'} = \max_{x, x'} g_{tt'}(x, x'). \tag{12}$$

The **height of a max-sum problem** $(G, X, \mathbf{g})$ is

$$U(\mathbf{g}) = \sum_t u_t + \sum_{\{t,t'\}} u_{tt'}. \tag{13}$$

Comparing corresponding terms in (5) and (13) yields that the problem height is an upper bound of quality, i.e., any $\mathbf{g}$ and any $\mathbf{x}$ satisfy $F(\mathbf{x} \,|\, \mathbf{g}) \leq U(\mathbf{g})$.

Unlike the quality function, the problem height is not invariant to equivalent transformations. This naturally leads to minimizing this upper bound by equivalent transformations, expressed by the linear program

$$U^*(\mathbf{g}) = \min_{\mathbf{g}' \sim \mathbf{g}} U(\mathbf{g}') \tag{14a}$$

$$= \min_{\boldsymbol{\varphi} \in \mathbb{R}^P} \Big[\sum_t \max_x g_t^{\boldsymbol{\varphi}}(x) + \sum_{\{t,t'\}} \max_{x, x'} g_{tt'}^{\boldsymbol{\varphi}}(x, x') \Big]. \tag{14b}$$

Remark 3: Some equivalent transformations preserve $U(\mathbf{g})$, e.g., adding a constant to all nodes of an object and subtracting the same constant from all nodes of another object. Thus, there may be many problems with the same height within every equivalence class. This gives an option to impose constraints on u_t and $u_{tt'}$ in the minimization and reformulate (14) in a number of ways, e.g.

$$U^*(\mathbf{g}) = \min_{\boldsymbol{\varphi} \in \mathbb{R}^P \,|\, g_{tt'}^{\boldsymbol{\varphi}}(x, x') \leq 0} \sum_t \max_x g_t^{\boldsymbol{\varphi}}(x) \tag{15a}$$

$$= |T| \min_{\boldsymbol{\varphi} \in \mathbb{R}^P \,|\, g_{tt'}^{\boldsymbol{\varphi}}(x, x') \leq 0} \max_t \max_x g_t^{\boldsymbol{\varphi}}(x). \tag{15b}$$

Form (15a) corresponds to imposing $u_{tt'} \leq 0$. Form (15b) corresponds to $u_{tt'} \leq 0$ and $u_t = u_{t'} = u$. Other natural constraints are $u_t = 0$, or $u_t = u_{t'} = u_{tt'}$.

D. Trivial Problems

Node (t, x) is a **maximal node** if $g_t(x) = u_t$. Edge $\{(t, x), (t', x')\}$ is a **maximal edge** if $g_{tt'}(x, x') = u_{tt'}$, where $\mathbf{u}$ is given by (12). Let this be expressed by Boolean variables

$$\bar{g}_t(x) = [\![g_t(x) = u_t]\!], \quad \bar{g}_{tt'}(x, x') = [\![g_{tt'}(x, x') = u_{tt'}]\!]. \tag{16}$$

A max-sum problem is **trivial** if a labeling can be formed of (some or all of) its maximal nodes and edges, i.e., if the CSP $(G, X, \bar{\mathbf{g}})$ with $\bar{\mathbf{g}}$ given by (16) is satisfiable. It is easy to see that the upper bound is tight, i.e. $F(\mathbf{x} \,|\, \mathbf{g}) = U(\mathbf{g})$ for some $\mathbf{x}$, for and only for trivial problems. This allows to formulate the following theorem, central to the whole approach.

Theorem 4: Let C be a class of equivalent max-sum problems. Let C contain a trivial problem. Then any problem in C is trivial if and only if its height is minimal in C.

Proof. Let $(G, X, \mathbf{g})$ be a trivial problem in C. Let a labeling $\mathbf{x}$ be composed of the maximal nodes and edges of $(G, X, \mathbf{g})$. Any $\mathbf{g}' \sim \mathbf{g}$ satisfies $U(\mathbf{g}') \geq F(\mathbf{x} \,|\, \mathbf{g}') = F(\mathbf{x} \,|\, \mathbf{g}) = U(\mathbf{g})$. Thus $(G, X, \mathbf{g})$ has minimal height.

Let $(G, X, \mathbf{g})$ be a non-trivial problem with minimal height in C. Any $\mathbf{g}' \sim \mathbf{g}$ and any optimal $\mathbf{x}$ satisfy $U(\mathbf{g}') \geq U(\mathbf{g}) > F(\mathbf{x} \,|\, \mathbf{g}) = F(\mathbf{x} \,|\, \mathbf{g}')$. Thus C contains no trivial problem. ■

Theorem 4 allows to divide the solution of a max-sum problem into two steps:

1) minimize the problem height by equivalent transformations,
2) test the resulting problem for triviality.

If the resulting problem with minimal height is trivial, i.e. $(G, X, \bar{\mathbf{g}})$ is satisfiable, then $L_{G,X}(\mathbf{g}) = \bar{L}_{G,X}(\bar{\mathbf{g}})$. If not, by theorem 4 the max-sum problem has no trivial equivalent and remains unsolved. In the former case the relaxation (8) is tight and in the latter case it is not.

Testing for triviality is NP-complete, equivalent to CSP. Thus, recognizing whether a given upper bound is tight is NP-complete. Even if we *knew* that a given upper bound $U(\mathbf{g})$ is tight, finding a labeling $\mathbf{x}$ such that $F(\mathbf{x} \,|\, \mathbf{g}) = U(\mathbf{g})$ still would be NP-complete. We can prove or disprove tightness of an upper bound only in special cases, such as those given by theorem 2.

Figure 3, giving examples of CSPs, can be interpreted also in terms of triviality if we imagine that the black nodes are maximal, the white nodes are non-maximal, and the shown edges are maximal. Then figure 3a shows a trivial problem (thus having minimal height), 3b a problem with a non-minimal height (hence non-trivial), and 3c a non-trivial problem with minimal height.

Note that not every polynomially solvable subclass of the max-sum problem has a trivial equivalent: e.g., if G is a simple

loop dynamic programming is applicable but figure 3c shows there might be no trivial equivalent.

E. Linear Programming Duality

The linear programs (8) and (14) are dual to each other [1, theorem 2]. To show this, we wrote them together in equation (11) (table I) such that a constraint and its Lagrange multiplier are on the same line, as is usual in linear programming.

The pair (11) can be slightly modified, corresponding to modifications of the primal constraints (7) and imposing constraints on dual variables $\mathbf{u}$, as discussed in remarks 1 and 3.

Duality of (8) and upper bound minimization was independently shown also by Wainwright *et al*. [15], [24] in the framework of convex combinations of trees. In our case, when the trees are objects and object pairs, proving the duality is more straightforward then for general trees.

Schlesinger and Kovalevsky [56] proposed elegant physical models of the pair (11). We described one of them in [41].

V. Conditions for Minimal Upper Bound

This section discusses how we can recognize that the height $U(\mathbf{g})$ of a max-sum problem is minimal among its equivalents, i.e., that $\mathbf{g}$ is optimal to (11). The main result will be that a non-empty kernel of the CSP formed by the maximal nodes and edges is necessary but not sufficient for minimal height.

To test for optimality of (11), linear programming duality theorems [57] give us a starting point. By weak duality, any $\mathbf{g}$ and any $\boldsymbol{\alpha} \in \Lambda_{G,X}$ satisfy $\langle \mathbf{g}, \boldsymbol{\alpha} \rangle \leq U(\mathbf{g})$. By strong duality, $\langle \mathbf{g}, \boldsymbol{\alpha} \rangle = U(\mathbf{g})$ if and only if $\mathbf{g}$ has minimal height and $\boldsymbol{\alpha}$ has maximal quality. By complementary slackness, $\langle \mathbf{g}, \boldsymbol{\alpha} \rangle = U(\mathbf{g})$ if and only if $\boldsymbol{\alpha}$ is zero on non-maximal nodes and edges.

To formalize the last statement, we define the **relaxed CSP** $(G, X, \bar{\mathbf{g}})$ as finding relaxed labelings on given nodes and edges, i.e., finding the set $\bar{\Lambda}_{G,X}(\bar{\mathbf{g}})$ of relaxed labelings $\boldsymbol{\alpha} \in \Lambda_{G,X}$ satisfying the complementarity constraints

$$[1 - \bar{g}_t(x)]\,\alpha_t(x) = 0, \quad [1 - \bar{g}_{tt'}(x, x')]\,\alpha_{tt'}(x, x') = 0. \quad (17)$$

Thus, $\bar{\Lambda}_{G,X}(\bar{\mathbf{g}})$ is the set of solutions to system (7)+(17). A CSP $(G, X, \bar{\mathbf{g}})$ is **relaxed-satisfiable** if $\bar{\Lambda}_{G,X}(\bar{\mathbf{g}}) \neq \emptyset$.

Further in this section, we let $\bar{\mathbf{g}}$ denote a function of $\mathbf{g}$ given by (16). In other words, $(G, X, \bar{\mathbf{g}})$ is not seen as an independent CSP but it is composed of the maximal nodes an edges of the max-sum problem $(G, X, \mathbf{g})$. Complementary slackness now reads as follows.

Theorem 5: The height of $(G, X, \mathbf{g})$ is minimal of all its equivalents if and only if $(G, X, \bar{\mathbf{g}})$ is relaxed-satisfiable. If it is so then $\Lambda_{G,X}(\mathbf{g}) = \bar{\Lambda}_{G,X}(\bar{\mathbf{g}})$.

A. Non-empty Kernel Necessary for Minimal Upper Bound

In section III, the concepts of arc consistency and kernel have been shown useful for characterizing CSP satisfiability. They are useful also for characterizing relaxed satisfiability. To show that, we first generalize the result that taking kernel preserves $\bar{L}_{G,X}(\bar{\mathbf{g}})$:

Theorem 6: Let $(G, X, \bar{\mathbf{g}}^*)$ be the kernel of a CSP $(G, X, \bar{\mathbf{g}})$. Then $\bar{\Lambda}_{G,X}(\bar{\mathbf{g}}) = \bar{\Lambda}_{G,X}(\bar{\mathbf{g}}^*)$.

Proof. Obvious from the argument in section III. A formal proof in [41]. ■

Thus, theorem 2 can be extended to relaxed labelings:

Theorem 7: A non-empty kernel of $(G, X, \bar{\mathbf{g}})$ is necessary for its relaxed satisfiability, hence for minimal height of $(G, X, \mathbf{g})$.

Proof. An immediate corollary of theorem 6. Alternatively, it is instructive to consider also the following dual proof.

We will denote the **height of pencil** (t, t', x) by $u_{tt'}(x) = \max_{x'} g_{tt'}(x, x')$ and call (t, t', x) a **maximal pencil** if it contains a maximal edge. Let us modify the arc consistency algorithm such that rather than by explicitly zeroing variables $\bar{\mathbf{g}}$ like in (4), nodes and edges of $(G, X, \bar{\mathbf{g}})$ are deleted by repeating the following equivalent transformations on $(G, X, \mathbf{g})$:

- Find a pencil (t, t', x) such that $u_{tt'}(x) < u_{tt'}$ and $g_t(x) = u_t$. Decrease node (t, x) by $\varphi_{tt'}(x) = \frac{1}{2}[u_{tt'} - u_{tt'}(x)]$. Increase all edges in pencil (t, t', x) by $\varphi_{tt'}(x)$.
- Find a pencil (t, t', x) such that $u_{tt'}(x) = u_{tt'}$ and $g_t(x) < u_t$. Increase node (t, x) by $\varphi_{tt'}(x) = \frac{1}{2}[u_t - g_t(x)]$. Decrease all edges in pencil (t, t', x) by $\varphi_{tt'}(x)$.

When no such pencil exists, the algorithm halts.

If the kernel of $(G, X, \bar{\mathbf{g}})$ was initially non-empty, the algorithm halts after the maximal nodes and edges that were not in the kernel are made non-maximal. If the kernel was initially empty, the algorithm sooner or later decreases the height of some node or edge, hence $U(\mathbf{g})$. ■

The algorithm in the proof has only a theoretical value. In practice, it is useless due to its slow convergence.

B. Non-empty Kernel Insufficient for Minimal Upper Bound

One might hope that non-empty kernel is not only necessary but also sufficient for relaxed satisfiability. Unfortunately, this is false, as was observed by Schlesinger [8] and, analogically in terms of convex combination of trees, by Kolmogorov [28], [30]. Figures 5b,c,d show counter-examples. We will justify these counter-examples first by giving a primal argument (i.e., by showing that $(G, X, \bar{\mathbf{g}})$ is not relaxed-satisfiable) then by giving a dual argument (i.e., by giving an equivalent transformation that decreases $U(\mathbf{g})$).

1) Primal argument: Let $(G, X, \bar{\mathbf{g}}^*)$ denote the kernel of a CSP $(G, X, \bar{\mathbf{g}})$. Consider an edge $\{(t, x), (t', x')\}$. By theorem 6, existence of $\boldsymbol{\alpha} \in \bar{\Lambda}_{G,X}(\bar{\mathbf{g}})$ such that $\alpha_{tt'}(x, x') > 0$ implies $\bar{g}^*_{tt'}(x, x') = 1$. Less obviously, the opposite implication is false. In other words, the fact that an edge belongs to the kernel is necessary but not sufficient for some relaxed labeling to be non-zero on this edge. The same holds for nodes. Figure 5a shows an example: it can be verified that system (7a)+(17) implies that $\alpha_{tt'}(x, x') = 0$ on the edge marked by the cross.

In figures 5b and 5c, the only solution to system (7a)+(17) is $\boldsymbol{\alpha} = \mathbf{0}$, therefore $\bar{\mathbf{g}}$ is relaxed-unsatisfiable. Note that 5b contains 5a as its part.

2) Dual argument: The analogical dual observation is that the kernel of $(G, X, \bar{\mathbf{g}})$ is not invariant to equivalent transformations of $(G, X, \mathbf{g})$. Consider the transformations in figure 5, depicted by non-zero values of $\varphi_{tt'}(x)$ written next to

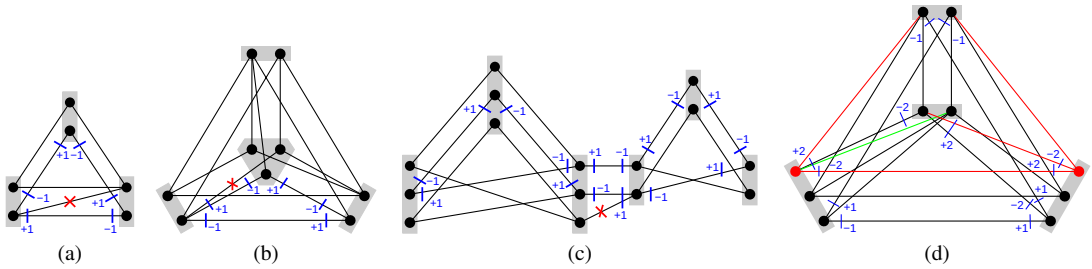


Fig. 5. Examples of kernels not invariant to equivalent transformations. The shown edges have quality 0 and the not shown edges $-\infty$. Problem (a) has minimal height, problems (b, c) do not; in particular, for (b, c) system (7a)+(7b)+(17) is unsolvable. For problem (d), system (7a)+(7b)+(17) is solvable but system (7a)+(7b)+(7c)+(17) is not.

the line segments crossing edge pencils (t, t', x). In each subfigure, the shown transformation makes the edge marked by the cross non-maximal and thus deletes it from the kernel. After this, the kernel in figure 5a still remains non-empty while the kernels in 5b and 5c become empty, as is verified by doing the arc consistency algorithm by hand. Thus, in 5b and 5c a non-empty kernel of $(G, X, \bar{\mathbf{g}})$ does not suffice for minimality of the height of $(G, X, \mathbf{g})$.

In figures 5b and 5c, system (7a)+(7b)+(17) has no solution, without even considering constraint (7c). Figure 5d shows a more advanced counter-example, where system (7a)+(7b)+(17) has a (single) solution but this solution violates (7c).

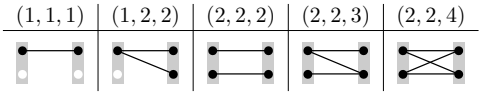
C. Boolean Max-sum Problems

For problems with Boolean variables ($|X| = 2$), Schlesinger observed [54] that a non-empty kernel is both necessary and sufficient for minimal upper bound. Independently, the equivalent observation was made by Kolmogorov and Wainwright [19], [31], who showed that weak tree agreement is sufficient for minimality of Wainwright's tree-based upper bound [24]. In addition, both noticed that for Boolean variables at least one relaxed labeling is half-integral; an analogical observation was made in pseudo-Boolean optimization [18], referring to [58].

Theorem 8: Let a CSP $(G, X, \bar{\mathbf{g}})$ with $|X| = 2$ labels have a non-empty kernel. Then $\bar{\Lambda}_{G,X}(\bar{\mathbf{g}}) \cap \{0, \frac{1}{2}, 1\}^I \neq \emptyset$.

Proof. We will prove the theorem by constructing a relaxed labeling $\boldsymbol{\alpha} \in \bar{\Lambda}_{G,X}(\bar{\mathbf{g}}) \cap \{0, \frac{1}{2}, 1\}^I$.

Delete all nodes and edges not in the kernel. Denote the number of nodes in object t and the number of edges in pair $\{t, t'\}$ by $n_t = \sum_x \bar{g}_t(x)$ and $n_{tt'} = \sum_{x,x'} \bar{g}_{tt'}(x, x')$, respectively. All object pairs can be partitioned into five classes (up to swapping labels), indexed by triplets $(n_t, n_{t'}, n_{tt'})$:



Remove one edge in each pair of class $(2, 2, 3)$ and two edges in each pair of class $(2, 2, 4)$ such that they become $(2, 2, 2)$. Now there are only pairs of classes $(1, 1, 1)$, $(1, 2, 2)$ and $(2, 2, 2)$. Let $\alpha_t(x) = \bar{g}_t(x)/n_t$ and $\alpha_{tt'}(x, x') = \bar{g}_{tt'}(x, x')/n_{tt'}$. Clearly, this $\boldsymbol{\alpha}$ belongs to $\bar{\Lambda}_{G,X}(\bar{\mathbf{g}})$. ■

For $|X| > 2$, a relaxed labeling that is an integer multiple of $|X|^{-1}$ may not exist. A counter-example is in figure 6.

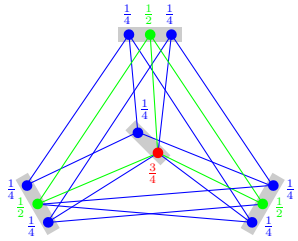


Fig. 6. A CSP for which $\bar{\Lambda}_{G,X}(\bar{\mathbf{g}})$ has a single element $\boldsymbol{\alpha}$ that is not an integer multiple of $|X|^{-1}$. This can be verified by solving system (7a)+(7b)+(17).

D. Summary: Three Kinds of Consistency

To summarize, we have met three kinds of 'consistency', related by implications as follows:

$$\begin{matrix}\bar{\mathbf{g}} \text{ satisfiable} \\ \mathbf{g} \text{ trivial}\end{matrix} \Rightarrow \begin{matrix}\bar{\mathbf{g}} \text{ relaxed-satisfiable} \\ \text{height of } \mathbf{g} \text{ minimal}\end{matrix} \Rightarrow \text{kernel of } \bar{\mathbf{g}} \text{ non-empty.}$$

The opposite implications do not hold in general. Exceptions are problems with two labels, for which non-empty kernel equals relaxed satisfiability, and supermodular max-sum problems (lattice CSPs) and problems on trees for which non-empty kernel equals satisfiability.

Testing for the first condition is NP-complete. Testing for the last condition is polynomial and simple, based on arc consistency. Testing for the middle condition is polynomial (solvable by linear programming) but we do not know any efficient algorithm to do this test for large instances. The difficulty seems to be in the fact that while arc consistency can be tested by local operations, relaxed satisfiability is probably an inherently non-local property.

To our knowledge, all known efficient algorithms for decreasing the height of max-sum problems use arc consistency or non-empty kernel as their termination criterion. We will review two such algorithms in sections VI and VII. Existence of arc consistent but relaxed-unsatisfiable configurations is unpleasant here because these algorithms need not find the minimal problem height. Analogical spurious minima occur also in the sequential tree-reweighted message passing (TRW-S) algorithm, as observed by Kolmogorov [27]–[30]. Omitting a formal proof, we argue that they are of the same nature as arc consistent relaxed-unsatisfiable states.

VI. Max-sum Diffusion

This section describes the max-sum diffusion algorithm [6], [7] to decrease the upper bound (13). It can be viewed as a coordinate descent method.

The diffusion is related to edge-based message passing by Wainwright [24, algorithm 1] but, unlike the latter, it is conjectured to always converge. Also, it can be viewed as the sequential tree-reweighted message passing (TRW-S) by Kolmogorov [27], [30], with the trees being nodes and edges (we omit a detailed proof). The advantage of the diffusion is its simplicity: it is even simpler than belief propagation.

A. The Algorithm

The **node-pencil averaging** on pencil (t,t',x) is the equivalent transformation that makes $g_t(x)$ and $u_{tt'}(x)$ equal, i.e., which adds number $\frac{1}{2}[u_{tt'}(x) - g_t(x)]$ to $g_t(x)$ and subtracts the same number from qualities of all edges in pencil (t,t',x). Recall that $u_{tt'}(x) = \max_{x'} g_{tt'}(x,x')$. In its simplest form, the max-sum diffusion algorithm repeats node-pencil averaging until convergence, on all pencils in any order such that each pencil is visited 'sufficiently often'. The following code does it (with a deterministic order of pencils):

repeat
  for $(t,t',x) \in P$ **do**
    $\varphi_{tt'}(x)$ += $\frac{1}{2}[\max_{x'} g^{\varphi}_{tt'}(x,x') - g^{\varphi}_t(x)]$;
  end for
until convergence
$\mathbf{g} := \mathbf{g}^{\varphi}$;

Remark 4: The algorithm can be easily made slightly more efficient. If a node (t,x) is fixed and node-pencil averaging is iterated on pencils $\{\,(t,t',x) \mid t' \in N_t\,\}$ till convergence, the heights of all these pencils and $g_t(x)$ become equal. This can be done by a single equivalent transformation on node (t,x).

B. Monotonicity

When node-pencil averaging is done on a single pencil, the problem height can decrease, remain unchanged, or increase. For an example when the height increases, consider a max-sum problem with $X = \{1,2\}$ such that for some pair $\{t,t'\}$, we have $g_t(1) = g_t(2) = 1$ and $u_{tt'}(1) = u_{tt'}(2) = -1$. After the node-pencil averaging on $(t,t',1)$, $U(\mathbf{g})$ increases by 1.

Monotonic height decrease can be ensured by choosing an appropriate order of pencils as given by theorem 9. This shows that the diffusion is a coordinate descent method.

Theorem 9: After the equivalent transformation consisting of $|X|$ node-pencil averagings on pencils $\{\,(t,t',x) \mid x \in X\,\}$, the problem height does not increase.

Proof. Before the transformation, the contribution of object t and pair $\{t,t'\}$ to $U(\mathbf{g})$ is $\max_x g_t(x) + \max_x u_{tt'}(x)$. After the transformation, this contribution is $\max_x [g_t(x) + u_{tt'}(x)]$. The first expression is not smaller than the second one because any two functions $f_1(x)$ and $f_2(x)$ satisfy $\max_x f_1(x) + \max_x f_2(x) \geq \max_x [f_1(x) + f_2(x)]$. ■

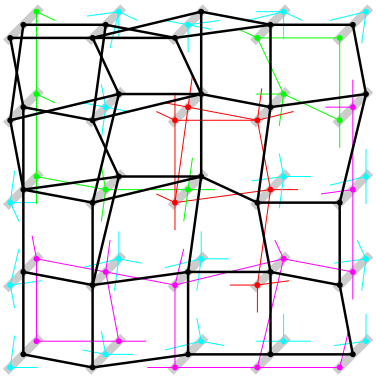


Fig. 7. A max-sum problem satisfying (18). A line segment starting from node (t,x) and aiming to but not reaching (t',x') denotes an edge satisfying $g_t(x) = g_{tt'}(x,x') < g_{t'}(x')$. If $g_t(x) = g_{tt'}(x,x') = g_{t'}(x')$, the line segment joins the nodes (t,x) and (t',x'). The grey levels help distinguish different layers; the highest layer is emphasized.

C. Properties of the Fixed Point

Based on numerous experiments, it was conjectured that the max-sum diffusion always converges. In addition, its fixed points can be characterized as follows:

Conjecture 1: For any $\mathbf{g} \in [-\infty,\infty)^I$, the max-sum diffusion converges to a solution of the system

$$\max_{x'} g_{tt'}(x,x') = g_t(x), \quad \{t,t'\} \in E,\ x \in X. \tag{18}$$

We are not aware of any proof of this conjecture.

Any solution to (18) has the following layered structure (see figure 7). A **layer** is a maximal connected subgraph of graph $(T \times X, E_X)$ such that its each edge $\{(t,x),(t',x')\}$ satisfies $g_t(x) = g_{tt'}(x,x') = g_{t'}(x')$. By (18), all nodes and edges of a layer have the same quality, the **height of the layer**. The highest layer is formed by the maximal nodes and edges.

Property (18) implies arc consistency of the maximal nodes and edges, as given by theorem 10. However, the converse is false: not every max-sum problem with arc consistent maximal nodes and edges satisfies (18).

Theorem 10: If a max-sum problem satisfies (18) then its maximal nodes and edges form an arc consistent CSP.

Proof. Suppose (18) holds. By (3), we are to prove that a pencil (t,t',x) is maximal if and only if node (t,x) is maximal. If (t,x) is maximal, then $u_{tt'}(x) = g_t(x) \geq g_t(x') = u_{tt'}(x')$ for each x', hence (t,t',x) is maximal. If (t,x) is non-maximal, then $u_{tt'}(x) = g_t(x) < g_t(x') = u_{tt'}(x')$ for some x', hence (t,t',x) is not maximal. ■

Since the max-sum problems in figures 5b,c,d satisfy (18), diffusion fixed points can have a non-minimal upper bound $U(\mathbf{g})$. This is a serious drawback of the algorithm.

More on max-sum diffusion can be found in the recent works [59], [60].

VII. The Augmenting DAG Algorithm

This section describes the height-decreasing algorithm given in [4], [9]. Its main idea is as follows: run the arc consistency

algorithm on the maximal nodes and edges, storing the pointers to the causes of deletions. When all nodes in a single object are deleted, it is clear that the kernel is empty. Backtracking the pointers provides a directed acyclic graph (DAG), called the augmenting DAG, along which a height-decreasing equivalent transformation is done.

The algorithm has been proved to converge in a finite number of steps [1] if it is modified as follows: maximality of nodes and edges is redefined using a threshold, ε. We will first explain the algorithm without this modification and return to it at the end of the section.

The iteration of the algorithm proceeds in three phases, described in subsequent sections. We use formulation (15a), i.e., we look for $\boldsymbol{\varphi}$ that minimizes $U(\mathbf{g}^{\boldsymbol{\varphi}}) = \sum_t \max_x g_t^{\boldsymbol{\varphi}}(x)$ subject to the constraint that all edges are non-positive, $g_{tt'}^{\boldsymbol{\varphi}}(x,x') \le 0$. Initially, all edges are assumed non-positive.

A. Phase 1: Arc Consistency Algorithm

The arc consistency algorithm is run on the maximal nodes and edges. It is not done exactly as described by rules (4) but in a slightly modified way as follows.

A variable $p_t(x) \in \{\texttt{ALIVE}, \texttt{NONMAX}\} \cup T$ is assigned to each node (t,x). Initially, we set $p_t(x) := \texttt{ALIVE}$ if (t,x) is maximal and $p_t(x) := \texttt{NONMAX}$ if (t,x) is non-maximal.

If a pencil (t,t',x) is found satisfying $p_t(x) = \texttt{ALIVE}$ and violating condition

$$(\exists x') \left[\begin{array}{c} \text{edge } \{(t,x),(t',x')\} \text{ is maximal,} \\ p_{t'}(x') = \texttt{ALIVE} \end{array} \right], \qquad (19)$$

node (t,x) is deleted by setting $p_t(x) := t'$. The object t' is called the **deletion cause** of node (t,x). This is repeated until either no such pencil exists, or an object t^* is found with $p_{t^*}(x) \ne \texttt{ALIVE}$ for all $x \in X$. In the former case, the augmenting DAG algorithm halts. In the latter case, we proceed to the next phase.

After every iteration of this algorithm, the maximal edges and the variables $p_t(x)$ define a directed acyclic subgraph D of graph $(T \times X, E_X)$, as follows: the nodes of D are the end nodes of its edges; edge $((t,x),(t',x'))$ belongs to D if and only if it is maximal and $p_t(x) = t'$. Once t^* has been found, the **augmenting DAG** $D(t^*)$ is a subgraph of D reachable by a directed path in D from the maximal nodes of t^*.

Example. The example max-sum problem in figure 8 has $T = \{a,\dots,f\}$ and the labels in each object are $1,2,3$, numbered from bottom to top. Figure 8a shows the maximal edges and the values of $p_t(x)$ after the first phase, when 10 nodes have been deleted by applying rule (19) successively on pencils $(c,a,2)$, $(c,a,3)$, $(e,c,1)$, $(e,c,3)$, $(f,e,3)$, $(d,c,2)$, $(b,d,2)$, $(a,b,2)$, $(d,b,1)$, $(f,d,1)$. The non-maximal edges are not shown. The nodes with $p_t(x) = \texttt{ALIVE}$ are small filled, with $p_t(x) = \texttt{NONMAX}$ small unfilled, and with $p_t(x) \in T$ large filled. For the deleted nodes, the causes $p_t(x)$ are denoted by short segments across pencils $(t, p_t(x), x)$. The object $t^* = f$ has a black outline.

Figure 8a shows D after t^* has been found and figure 8b shows $D(t^*)$. The edges of D and $D(t^*)$ are emphasized and the nodes, except $(a,3)$, too. ■

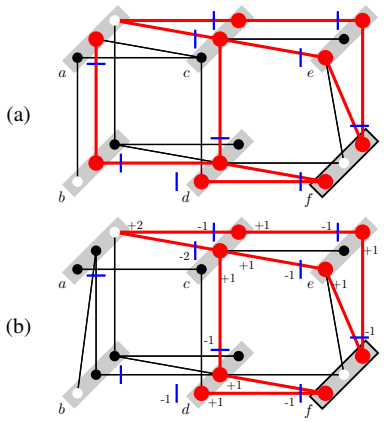


Fig. 8. (a) The augmenting DAG algorithm after phase 1; (b) after phase 2.

B. Phase 2: Finding the Search Direction

The direction of height decrease is found in the space $\mathbb{R}^P$, i.e., a vector $\Delta\boldsymbol{\varphi}$ is found such that $U(\mathbf{g}^{\boldsymbol{\varphi}+\lambda\Delta\boldsymbol{\varphi}}) < U(\mathbf{g}^{\boldsymbol{\varphi}})$ for a small positive λ.

Denoting $\Delta\varphi_t(x) = \sum_{t' \in N_t} \Delta\varphi_{tt'}(x)$, the vector $\Delta\boldsymbol{\varphi}$ has to satisfy

$$\begin{aligned} -\Delta\varphi_{t^*}(x) &= 1 \quad \text{if } p_{t^*}(x) \ne \texttt{NONMAX}, \\ \Delta\varphi_t(x) &\le 0 \quad \text{if } p_t(x) \ne \texttt{NONMAX}, \\ \Delta\varphi_{tt'}(x) + \Delta\varphi_{t't}(x') &\ge 0 \quad \text{if } \{(t,x),(t',x')\} \text{ maximal.} \end{aligned}$$

We find the smallest vector $\Delta\boldsymbol{\varphi}$ satisfying these. This is done by traversing $D(t^*)$ from roots to leaves, successively enforcing these constraints for all its nodes and edges. The traversal is done in a linear order on $D(t^*)$, i.e., a node is not visited before the tails of all edges entering it have been visited. In figure 8b, the non-zero numbers $\Delta\varphi_{tt'}(x)$ are written near their pencils.

C. Phase 3: Finding the Search Step

The search step length λ is found such that no edge becomes positive, the height of no object is increased, and the height of t^* is minimized. These read respectively

$$\begin{aligned} g_{tt'}^{\boldsymbol{\varphi}+\lambda\Delta\boldsymbol{\varphi}}(x,x') &\le 0, \\ g_t^{\boldsymbol{\varphi}+\lambda\Delta\boldsymbol{\varphi}}(x) &\le \max_x g_t^{\boldsymbol{\varphi}}(x), \\ g_{t^*}^{\boldsymbol{\varphi}+\lambda\Delta\boldsymbol{\varphi}}(x) &\le \max_x g_{t^*}^{\boldsymbol{\varphi}}(x) - \lambda. \end{aligned}$$

To justify the last inequality, see that each node of t^* with $p_{t^*}(x) \in T$ decreases by λ and each node with $p_{t^*}(x) = \texttt{NONMAX}$ increases by $\lambda\,\Delta\varphi_{t^*}(x)$. The latter is because $D(t^*)$ can have a leaf in t^*. To minimize the height of t^*, the nodes with $p_{t^*}(x) = \texttt{NONMAX}$ must not become higher than the nodes with $p_{t^*}(x) \in T$.

Solving the above three conditions for λ yields the system

$$\begin{aligned} \lambda &\le \frac{g_{tt'}^{\boldsymbol{\varphi}}(x,x')}{\Delta\varphi_{tt'}(x) + \Delta\varphi_{t't}(x')} \quad \text{if } \Delta\varphi_{tt'}(x) + \Delta\varphi_{t't}(x') < 0, \\ \lambda &\le \frac{u_t - g_t^{\boldsymbol{\varphi}}(x)}{[\![t = t^*]\!] + \Delta\varphi_t(x)} \quad \text{if } [\![t = t^*]\!] + \Delta\varphi_t(x) > 0. \end{aligned}$$

We find the greatest λ satisfying these.

The iteration of the augmenting DAG algorithm is completed by the equivalent transformation $\boldsymbol{\varphi} \mathrel{+}= \lambda \Delta\boldsymbol{\varphi}$.

For implementation details, refer to [4], [41].

The algorithm sometimes spends a lot of iterations to minimize the height in a subgraph of G accurately [61]. This is wasteful because this accuracy is destroyed once the subgraph is left. This behavior, somewhat similar to the well-known inefficiency of the Ford-Fulkerson max-flow algorithm, can be reduced by redefining maximality of nodes and edges using a threshold $\varepsilon > 0$ as follows [4]: node (t, x) is maximal if and only if $-g^{\boldsymbol{\varphi}}_{tt'}(x, x') \le \varepsilon$, and edge $\{(t, x), (t', x')\}$ is maximal if and only if $-g^{\boldsymbol{\varphi}}_{tt'}(x, x') \le \varepsilon$. If ε is reasonably large, 'nearly maximal' nodes and edges are considered maximal and often a larger λ results. With $\varepsilon > 0$, the algorithm terminates in finite number of iterations [4]. A possible scheme is to run the algorithm several times, exponentially decreasing ε.

Since they are arc-consistent, the problems in figures 5b,c,d are termination states of the algorithm. Thus, the algorithm can terminate with a non-minimal upper bound $U(\mathbf{g})$.

VIII. Supermodular Max-sum Problems

(Super-) submodularity, for bivariate functions also known as *(inverse) Monge property* [62], is well-known to simplify many optimization tasks; in fact, it can be considered a discrete counterpart of convexity [63]. It is long known that set supermodular max-sum problems can be translated to max-flow/min-cut [17], [64] and therefore are tractable. Some authors suggested this independently, e.g. Kolmogorov and Zabih [36]. Others showed translation to max-flow for other subclasses of the supermodular max-sum problem: Greig *et al.* [65]; Ishikawa and Geiger [66], [67] for the bivariate functions being convex univariate functions of differences of variable pairs; Cohen *et al.* for Max CSP [34]. D. Schlesinger and Flach [68] gave the translation to max-flow of the full class of supermodular max-sum problems; importantly, this is a special case of the more results that max-sum problem with any number of labels can be transformed to a problem with two labels [68]. In many of these works, especially in computer vision, connection with supermodularity was not noticed and the property was given *ad hoc* names.

Tractability of supermodular max-sum problems follows from a more general result. Their objective function is a special case of a supermodular function on a product of chains, which is in turn a special case of a supermodular function on a distributive lattice. Submodular functions with Boolean variables can be minimized in polynomial time [69], [70] and for submodular functions on distributive lattices, even strongly polynomial algorithms exist [71], [72].

Linear programming relaxation (11) was shown tight for supermodular problems by Schlesinger and Flach [11] and, independently, for Boolean supermodular problems by Kolmogorov and Wainwright [19], [31] using convex combination of trees [15], [24]. Further in this section, we prove this, following [11].

In particular, we will prove that if the function $F(\mathbf{x} \,|\, \mathbf{g})$ (or, equivalently, the functions $g_{tt'}(\bullet, \bullet)$) is supermodular then the max-sum problem has a trivial equivalent and finding an optimal labeling is tractable. We will proceed in two steps: first, we'll show that a certain subclass of CSP is tractable and, moreover, satisfiable if its kernel is non-empty; second, we'll show that the maximal nodes and edges of a supermodular problem always form a CSP in this subclass.

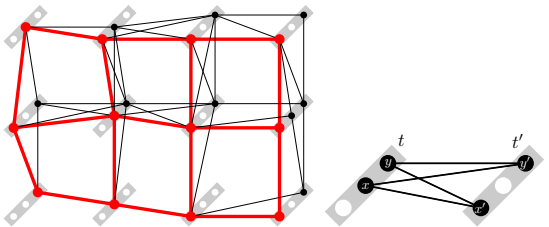


Fig. 9. (a) An arc consistent lattice CSP is always satisfiable because a labeling on it can be found by picking the lowest label in each object separately (emphasized). (b) Supermodular max-sum problems satisfy $g_{tt'}(x, x') + g_{tt'}(y, y') \ge g_{tt'}(x, y') + g_{tt'}(y, x')$ for every $x \le y$ and $x' \le y'$. It follows that the poset $\bar{L}_{tt'} = \{\, (x, x') \mid g_{tt'}(x, x') = u_{tt'} \,\}$ is a lattice. In the pictures, the order $\le$ is given by the vertical direction.

We assume that the label set X is endowed with a (known) total order $\le$, i.e., the poset $(X, \le)$ is a chain. The product $(X^n, \le)$ of n these chains is a distributive lattice, with the new partial order given componentwise, and with meet $\wedge$ (join $\vee$) being componentwise minimum (maximum). In this section, $\wedge$ and $\vee$ denote meet and join rather than logical conjunction and disjunction. See [41], [55], [73] for background on lattices and supermodularity.

Let $\bar{L}_{tt'} = \{\, (x, x') \mid \bar{g}_{tt'}(x, x') = 1 \,\}$. We call $(G, X, \bar{\mathbf{g}})$ a **lattice CSP** if the poset $(\bar{L}_{tt'}, \le)$ is a lattice (i.e., is closed under meet and join) for every $\{t, t'\} \in E$. Note, it follows easily that for a lattice CSP, $\bar{L}_{G,X}(\bar{\mathbf{g}})$ is also a lattice. Theorem 11 shows that lattice CSPs are tractable.

Theorem 11: Any arc consistent lattice CSP $(G, X, \bar{\mathbf{g}})$ is satisfiable. The 'lowest' labeling $\mathbf{x} = \bigwedge \bar{L}_{G,X}(\bar{\mathbf{g}})$ is given by $x_t = \min\{\, x \in X \mid \bar{g}_t(x) = 1 \,\}$ (x_t are the components of $\mathbf{x}$).

Proof. It is obvious from figure 9a that the 'lowest' nodes and edges form a labeling. Here is a formal proof.

Let $x_t = \min\{\, x \in X \mid \bar{g}_t(x) = 1 \,\}$. We'll show that $\bar{g}_{tt'}(x_t, x_{t'}) = 1$ for $\{t, t'\} \in E$. Pick $\{t, t'\} \in E$. By (3), pencil (t, t', x_t) contains at least one edge, while pencils $\{\, (t, t', x) \mid x < x_t \,\}$ are empty. Similarly for pencils $(t', t, x_{t'})$ and $\{\, (t', t, x') \mid x' < x_{t'} \,\}$. Since $(\bar{L}_{tt'}, \le)$ is a lattice, the meet of the edges in pair $\{t, t'\}$ is $\{(t, x_t), (t', x_{t'})\}$. ∎

Recall that a function $f\colon A \to \mathbb{R}$ on a lattice $(A, \le)$ is **supermodular** if all $a, b \in A$ satisfy

$$f(a \wedge b) + f(a \vee b) \ge f(a) + f(b). \tag{20}$$

In particular, a bivariate function f (i.e., $(A, \le)$ is a product of two chains, $(X^2, \le)$) is supermodular if and only if $x \le y$ and $x' \le y'$ implies $f(x, x') + f(y, y') \ge f(x, y') + f(y, x')$.

We say $(G, X, \mathbf{g})$ is a **supermodular max-sum problem** if all the functions $g_{tt'}(\bullet, \bullet)$ are supermodular on $(X^2, \le)$. The following theorem shows that this is equivalent to supermodularity of the function $F(\bullet \,|\, \mathbf{g})$.

Theorem 12: The function $F(\bullet \,|\, \mathbf{g})$ is supermodular if and only if all the bivariate functions $g_{tt'}(\bullet, \bullet)$ are supermodular.

Proof. The *if* part is true because by (20), a sum of supermodular function is supermodular.

The *only if* part. Pick a pair $\{t, t'\}$. Let two labelings $\mathbf{x}, \mathbf{y} \in X^T$ be equal in all objects except t and t' where they satisfy $x_t \leq x_{t'}$ and $y_t \geq y_{t'}$. If $F(\bullet \,|\, \mathbf{g})$ is supermodular, by (20) it is $F(\mathbf{x} \wedge \mathbf{y} \,|\, \mathbf{g}) + F(\mathbf{x} \vee \mathbf{y} \,|\, \mathbf{g}) \geq F(\mathbf{x} \,|\, \mathbf{g}) + F(\mathbf{y} \,|\, \mathbf{g})$. After substitution from (5) and some manipulations, we are left with $g_{tt'}(x_t, y_{t'}) + g_{tt'}(y_t, x_{t'}) \geq g_{tt'}(x_t, x_{t'}) + g_{tt'}(y_t, y_{t'})$. ■

Function $F(\bullet \,|\, \mathbf{g})$ is invariant to equivalent transformations. Theorem 12 implies that supermodularity of $g_{tt'}(\bullet, \bullet)$ is so too. This is also seen from the fact that an equivalent transformation means adding a zero problem, which is modular, and supermodularity is preserved by adding a modular function.

The following theorem shows that the maximal nodes and edges of a supermodular problem form a lattice CSP.

Theorem 13: [55] The set A^* of maximizers of a supermodular function f on a lattice A is a sublattice of A.

Proof. Let $a, b \in A^*$. Denote $p = f(a) = f(b)$, $q = f(a \wedge b)$, and $r = f(a \vee b)$. Maximality of p implies $p \geq q$ and $p \geq r$. Supermodularity condition $q + r \geq 2p$ yields $p = q = r$. ■

The theorem can be applied to function f being either $g_{tt'}(\bullet, \bullet)$ or $F(\bullet \,|\, \mathbf{g})$. This completes the proof that every supermodular max-sum problem has a trivial equivalent and is tractable.

IX. Application to Structural Image Analysis

Even if this article primarily focuses on theory, we present an example of applying the approach to structural image analysis. It is motivated by those in [1], [9] and we give more such examples in [41]. The task is different from non-supermodular problems of Potts type and arising from stereo reconstruction, experimentally examined in [19], [28], [29], [31], [74], [75], in the fact that a lot of edge qualities are $-\infty$. In that, our example is closer to CSP. In the sense of [1], [9], it can be interpreted as finding the 'nearest' image belonging to the language generated by a given 2D grammar (in full generality, 2D grammars include also hidden variables). If qualities are viewed as log-likelihoods, the task corresponds to finding the maximum of a Gibbs distribution.

Let the following be given. Let G represent a 4-connected image grid. Each pixel $t \in T$ has a label from $X = \{\mathtt{E}, \mathtt{I}, \mathtt{T}, \mathtt{L}, \mathtt{R}\}$. Numbers $g_{tt'}(x, x')$ are given by figure 10a, which shows three pixels forming one horizontal and one vertical pair, as follows: the solid edges have quality 0, the dashed edges $-\frac{1}{2}$, and the edges not shown $-\infty$. The functions $g_{tt'}(\bullet, \bullet)$ for all vertical pairs are equal, as well as for all horizontal pairs.

Numbers $f(\mathtt{E}) = f(\mathtt{I}) = 1$ and $f(\mathtt{T}) = f(\mathtt{L}) = f(\mathtt{R}) = 0$ assign an intensity to each label. Thus, $\mathbf{f}(\mathbf{x}) = \{\, f(x_t) \mid t \in T \,\}$ is the black-and-white image corresponding to labeling $\mathbf{x}$.

First, assume that $g_t(x) = 0$ for all t and x. The set $\{\, \mathbf{f}(\mathbf{x}) \mid F(\mathbf{x} \,|\, \mathbf{g}) > -\infty \,\}$ contains images feasible to the 2D grammar $(G, X, \mathbf{g})$, here, images of multiple non-overlapping black 'free-form' characters 'Π' on white background. An example of such an image with labels denoted is in figure 10b. The number of characters in the image is $-F(\mathbf{x} \,|\, \mathbf{g})$.

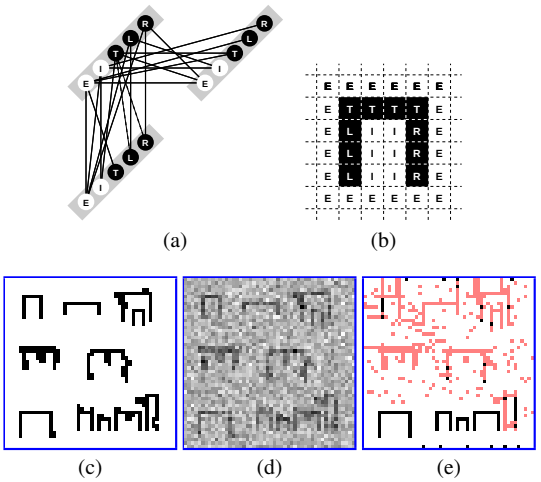


Fig. 10. The 'Letters Π' example. (a) The vertical and horizontal pixel pair defining the problem. (b) A labeled image feasible to this definition. The input image in (d) is the image in (c) plus independent Gaussian noise. (e) The output image. Image size 50×50 pixels.

Let an input image $\{\, f_t \mid t \in T \,\}$ be given. The numbers $g_t(x) = -c\,[f_t - f(x)]^2$ quantify similarity between the input image and the intensities of the labels; we set $c = \frac{1}{6}$. Setting the dashed edges in figure 10a to a non-zero value discourages images with a large number of small characters, which can be viewed as a regularization.

For the input in figure 10d, we minimized the height of the max-sum problem $(G, X, \mathbf{g})$ by the augmenting DAG algorithm and then computed the kernel of the CSP formed by the maximal nodes and edges. To get a partial and suboptimal solution to the CSP, we used the unique label condition from theorem 2. The result is in figure 10e. A pixel t with a unique maximal node (t, x) is black or white as given by $f(x)$, a pixel with multiple maximal nodes is gray. Unfortunately, there are rather many ambiguous pixels.

It turns out that if X and $\mathbf{g}$ are redefined by adding two more labels as shown in figure 11, a unique label in each pixel is obtained. We observed this repeatedly: of several formulations of the max-sum problem defining the same feasible set $\{\, \mathbf{f}(\mathbf{x}) \mid F(\mathbf{x} \,|\, \mathbf{g}) > -\infty \,\}$, some (usually not the simplest ones) provide tight upper bounds more often.

For figure 10, the runtime of the augmenting DAG algorithm (the implementation [41]) was 1.6 s on a 1.2 GHz laptop PC, and the max-sum diffusion achieved the state with arc consistent maximal nodes and edges in almost 8 min (maximality threshold 10^{-6}, double arithmetic). For figure 11, the augmenting DAG algorithm took 0.3 s and the diffusion 20 s.

X. Conclusion

We have reviewed the approach to the max-sum problem by Schlesinger *et al.* in a unified and self-contained framework.

The fact that due to non-optimal fixed points, no efficient algorithm to minimize the upper bound $U(\mathbf{g})$ is known is the *most serious open question*. This is not only a gap in theory but

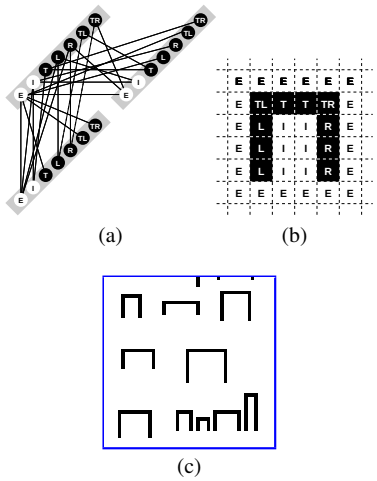


Fig. 11. The 'Letters Π 2' example, alternative 'better' definition of 'Letters Π'. (a) Definition, (b) a feasible labeled image, (c) output. The input was figure 10d.

also relevant in applications because the difference between the true and a spurious minimum can be arbitrarily large.

To present the approach by Schlesinger *et al*. in a single article, we had to omit some issues for lack of space. We have omitted a detailed formal comparison with the work by Wainwright *et al*. and Kolmogorov [19], [24], [30]. We have not discussed relation to other continuous relaxations [20]–[22], [76], to α-expansions and $\alpha\beta$-swaps [77], and to primal-dual schema [78]. We have not done experimental comparison of the max-sum diffusion and the augmenting DAG algorithms with other approximative algorithms for the max-sum problem [75], [79], [80]. We have not discussed persistency (partial optimality) results by Kolmogorov and Wainwright [19] for Boolean variables and by Kovtun [39], [40] for the (NP-hard) Potts model.

ACKNOWLEDGMENT

My work has been supported by the the European Union, grant IST-2004-71567. The article could not be written without Václav Hlaváč, who established co-operation of our group with the Kiev and Dresden groups in 1996 and has been supporting it since then, and my personal communication with Mikhail I. Schlesinger. Christoph Schnör, Alexander Shekhovtsov, Václav Hlaváč, Mirko Navara, Tomáš Pajdla, Jiří Matas, and Vojtěch Franc gave me valuable comments.

REFERENCES

[1] M. I. Schlesinger, "Sintaksicheskiy analiz dvumernykh zritelnikh signalov v usloviyakh pomekh (Syntactic analysis of two-dimensional visual signals in noisy conditions)," *Kibernetika*, vol. 4, pp. 113–130, 1976, in Russian.

[2] M. L. Minsky and S. A. Papert, *Perceptrons: An Introduction to Computational Geometry*, 2nd ed. Cambridge, MA, USA: MIT Press, 1988, first edition in 1971.

[3] A. Mackworth, "Constraint satisfaction," in *Encyclopedia of Artificial Intelligence*. New York: Wiley, 1991, pp. 285–292.

[4] V. K. Koval and M. I. Schlesinger, "Dvumernoe programmirovanie v zadachakh analiza izobrazheniy (Two-dimensional programming in image analysis problems)," *USSR Academy of Science, Automatics and Telemechanics*, vol. 8, pp. 149–168, 1976, in Russian.

[5] V. A. Kovalevsky, M. I. Schlesinger, and V. K. Koval, "Ustrojstvo dlya analiza seti," Patent Nr. 576843, USSR, priority of January 4, 1976, 1977, in Russian.

[6] V. A. Kovalevsky and V. K. Koval, "A diffusion algorithm for decreasing energy of max-sum labeling problem," approx. 1975, Glushkov Institute of Cybernetics, Kiev, USSR. Unpublished.

[7] B. Flach, "A diffusion algorithm for decreasing energy of max-sum labeling problem," 1998, Fakultät Informatik, Technische Universität Dresden, Germany. Unpublished.

[8] M. I. Schlesinger, "False minima of the algorithm for minimizing energy of max-sum labeling problem," 1976, Glushkov Institute of Cybernetics, Kiev, USSR. Unpublished.

[9] ——, *Matematicheskie sredstva obrabotki izobrazheniy (Mathematical Tools of Image Processing)*. Naukova Dumka, Kiev, 1989, in Russian.

[10] M. I. Schlesinger and V. Hlaváč, *Ten Lectures on Statistical and Structural Pattern Recognition*, M. A. Viergever, Ed. Dordrecht, The Netherlands: Kluwer Academic Publishers, 2002.

[11] M. I. Schlesinger and B. Flach, "Some solvable subclasses of structural recognition problems," in *Czech Patt. Recog. Workshop*, 2000.

[12] A. Koster, C. P. M. van Hoesel, and A. W. J. Kolen, "The partial constraint satisfaction problem: Facets and lifting theorems," *Operations Research Letters*, vol. 23, no. 3–5, pp. 89–97, 1998.

[13] A. Koster, "Frequency assignment – models and algorithms," Ph.D. dissertation, Universiteit Maastricht, Maastricht, The Netherlands, 1999, ISBN 90-9013119-1.

[14] C. Chekuri, S. Khanna, J. Naor, and L. Zosin, "Approximation algorithms for the metric labeling problem via a new linear programming formulation," in *Symposium on Discrete Algorithms*, 2001, pp. 109–118.

[15] M. Wainwright, T. Jaakkola, and A. Willsky, "MAP estimation via agreement on (hyper)trees: message passing and linear programming approaches," in *Allerton Conf. on Communication, Control and Computing*, 2002.

[16] C. L. Kingsford, B. Chazelle, and M. Singh, "Solving and analyzing side-chain positioning problems using linear and integer programming," *Bioinformatics*, vol. 21, no. 7, pp. 1028–1039, 2005.

[17] E. Boros and P. L. Hammer, "Pseudo-Boolean optimization," *Discrete Applied Mathematics*, vol. 123, no. 1-3, pp. 155–225, 2002.

[18] P. L. Hammer, P. Hansen, and B. Simeone, "Roof duality, complementation and persistency in quadratic 0-1 optimization," *Math. Programming*, vol. 28, pp. 121–155, 1984.

[19] V. N. Kolmogorov and M. J. Wainwright, "On the optimality of tree-reweighted max-product message-passing," in *Conf. Uncertainty in Artificial Intelligence (UAI)*, 2005.

[20] T. Wierschin and S. Fuchs, "Quadratic minimization for labeling problems," Technical University Dresden, Germany, Tech. Rep., 2002.

[21] P. Ravikumar and J. Lafferty, "Quadratic programming relaxations for metric labeling and Markov random field MAP estimation," in *Intl. Conf. Machine Learning ICML*, 2006.

[22] M. J. Wainwright and M. I. Jordan, "Semidefinite relaxations for approximate inference on graphs with cycles." in *Conf. Neural Information Processing Systems (NIPS)*, 2003.

[23] M. J. Wainwright, T. Jaakkola, and A. S. Willsky, "A new class of upper bounds on the log partition function," *IEEE Trans. Information Theory*, vol. 51, no. 7, pp. 2313–2335, 2005.

[24] M. Wainwright, T. Jaakkola, and A. Willsky, "MAP estimation via agreement on (hyper)trees: message passing and linear programming approaches," *IEEE Trans. Information Theory*, vol. 51, no. 11, pp. 3697–3717, 2005.

[25] ——, "Tree-based reparameterization framework for analysis of sum-product and related algorithms," *IEEE Trans. Information Theory*, vol. 49, no. 5, pp. 1120–1146, 2003.

[26] ——, "Tree consistency and bounds on the performance of the max-product algorithm and its generalizations," *Statistics and Computing*, vol. 14, pp. 143–166, 2004.

[27] V. Kolmogorov, "Convergent tree-reweighted message passing for energy minimization," Microsoft Research, Tech. Rep. MSR-TR-2004-90, 2004.

[28] ——, "Convergent tree-reweighted message passing for energy minimization," Microsoft Research, Tech. Rep. MSR-TR-2005-38, 2005.

[29] ——, "Convergent tree-reweighted message passing for energy minimization," in *Intl. Workshop on Artificial Intelligence and Statistics (AISTATS)*, 2005.

[30] ——, "Convergent tree-reweighted message passing for energy minimization," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 28, no. 10, pp. 1568–1583, 2006.

[31] V. Kolmogorov and M. Wainwright, "On the optimality of tree-reweighted max-product message-passing," Microsoft Research, Tech. Rep. MSR-TR-2004-37, 2005.

[32] J. Pearl, *Probabilistic reasoning in intelligent systems: Networks of plausible inference*. San Francisco: Morgan Kaufmann, 1988.

[33] J. S. Yedidia, W. T. Freeman, and Y. Weiss, "Constructing free-energy approximations and generalized belief propagation algorithms," *IEEE Trans. Information Theory*, vol. 51, no. 7, pp. 2282–2312, 2005.

[34] D. Cohen, M. Cooper, P. Jeavons, and A. Krokhin, "Supermodular functions and the complexity of Max CSP." *Discrete Applied Mathematics*, vol. 149, no. 1-3, pp. 53–72, 2005.

[35] S. Bistarelli, U. Montanari, F. Rossi, T. Schiex, G. Verfaillie, and H. Fargier, "Semiring-based CSPs and valued CSPs: Frameworks, properties,and comparison," *Constraints*, vol. 4, no. 3, pp. 199–240, 1999.

[36] V. Kolmogorov and R. Zabih, "What energy functions can be minimized via graph cuts?" in *European Conf. Computer Vision (ECCV)*. Springer-Verlag, 2002, pp. 65–81.

[37] D. Schlesinger, "Strukturelle ansätze für die stereorekonstruktion," Ph.D. dissertation, Technische Universität Dresden, Fakultät Informatik, Institut für Künstliche Intelligenz, July 2005, in German.

[38] B. Flach, "Strukturelle bilderkennung," Fakultät Informatik, Technische Universität Dresden, Germany, Tech. Rep., 2002, habilitation thesis, in German.

[39] I. Kovtun, "Partial optimal labelling search for a NP-hard subclass of (max,+) problems," in *Conf. German Assoc. for Pattern Recognition (DAGM)*, 2003, pp. 402–409.

[40] ——, "Segmentaciya zobrazhen na usnovi dostatnikh umov optimalnosti v NP-povnikh klasakh zadach strukturnoi rozmitki (Image segmentation based on sufficient conditions of optimality in NP-complete classes of structural labeling problems)," Ph.D. dissertation, IRTC ITS Nat. Academy of Science Ukraine, Kiev, 2004, in Ukrainian.

[41] T. Werner, "A linear programming approach to max-sum problem: A review," Center for Machine Perception, Czech Technical University, Tech. Rep. CTU–CMP–2005–25, December 2005.

[42] S. Verdú and H. V. Poor, "Abstract dynamic programming models under commutativity conditions," *SIAM J. Control and Optimization*, vol. 25, no. 4, pp. 990–1006, July 1987.

[43] S. Bistarelli, U. Montanari, and F. Rossi, "Semiring-based constraint satisfaction and optimization," *J. of ACM*, vol. 44, no. 2, pp. 201–236, 1997.

[44] S. Gaubert, "Methods and applications of (max,+) linear algebra," Institut national de recherche en informatique et en automatique (INRIA), Tech. Rep. 3088, 1997.

[45] S. M. Aji and R. J. McEliece, "The generalized distributive law," *IEEE Trans. on Information Theory*, vol. 46, no. 2, pp. 325–343, 2000.

[46] D. L. Waltz, "Generating semantic descriptions from drawings of scenes with shadows," Massachusetts Institute of Technology, Tech. Rep., 1972.

[47] U. Montanari, "Networks of constraints: Fundamental properties and application to picture processing," *Information Science*, vol. 7, pp. 95–132, 1974.

[48] A. Rosenfeld, R. A. Hummel, and S. W. Zucker, "Scene labeling by relaxation operations," *IEEE Trans. on Systems, Man, and Cybernetics*, vol. 6, no. 6, pp. 420–433, June 1976.

[49] A. K. Mackworth, "Consistency in networks of relations," *Artificial intelligence*, vol. 8, no. 1, pp. 65–73, 1977.

[50] R. M. Haralick and L. G. Shapiro, "The consistent labeling problem," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 1, no. 2, pp. 173–184, 1979.

[51] M. Grohe and D. Marx, "Constraint solving via fractional edge covers," in *Proc. the 17th annual ACM-SIAM symp. Discrete algorithm (SODA)*. ACM Press, 2006, pp. 289–298.

[52] A. Bulatov, P. Jeavons, and A. Krokhin, "Classifying the complexity of constraints using finite algebras," *Computing*, vol. 34, no. 3, pp. 720–742, 2005.

[53] R. Debruyne and C. Bessière, "Domain filtering consistencies," *Journal of Artificial Intelligence Research*, no. 14, pp. 205–230, May 2001.

[54] M. I. Schlesinger, "Lectures on labeling problems attended by the authors, Kiev, Prague, Dresden," 1996-2006.

[55] D. M. Topkis, "Minimizing a submodular function on a lattice," *Operations Research*, vol. 26, no. 2, pp. 305–321, 1978.

[56] M. I. Schlesinger and V. Kovalevsky, "A hydraulic model of a linear programming relaxation of max-sum labeling problem," 1978, Glushkov Institute of Cybernetics, Kiev, USSR. Unpublished.

[57] R. J. Vanderbei, *Linear Programming: Foundations and Extensions*. Boston: Kluwer Academic Publishers, 1996.

[58] M. L. Balinski, "Integer programming: methods, uses, computation," *Management Science*, vol. 12, no. 3, pp. 253–313, 1965.

[59] T. Werner and A. Shekhovtsov, "Unified framework for semiring-based arc consistency and relaxation labeling," in *12th Computer Vision Winter Workshop, St. Lambrecht, Austria*, M. Grabner and H. Grabner, Eds. Graz University of Technology, February 2007, pp. 27–34.

[60] T. Werner, "What is decreased by the max-sum arc consistency algorithm?" in *Intl. Conf. on Machine Learning, Oregon, USA*, June 2007.

[61] M. I. Schlesinger, "Personal communication," 2000-2005, International Research and Training Centre, Kiev, Ukraine.

[62] R. E. Burkard, B. Klinz, and R. Rudolf, "Perspectives of Monge properties in optimization," *Discrete Applied Math.*, vol. 70, no. 2, pp. 95–161, 1996.

[63] L. Lovász, "Submodular functions and convexity," in *Mathematical Programming – The State of the Art*, A. Bachem, M. Grötschel, and B. Korte, Eds. Springer-Verlag, New York, 1983, pp. 235–257.

[64] P. L. Hammer, "Some network flow problems solved with pseudo-Boolean programming," *Operations Research*, vol. 13, pp. 388–399, 1965.

[65] D. Greig, B. Porteous, and A. Seheult, "Exact maximum a posteriori estimation for binary images," *J. R. Statist. Soc. B*, no. 51, pp. 271–279, 1989.

[66] H. Ishikawa and D. Geiger, "Segmentation by grouping junctions," in *IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 1998, pp. 125–131.

[67] H. Ishikawa, "Exact optimization for Markov random fields with convex priors." *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 25, no. 10, pp. 1333–1336, 2003.

[68] D. Schlesinger and B. Flach, "Transforming an arbitrary MinSum problem into a binary one," Dresden University of Technology, Germany, Tech. Rep. TUD-FI06-01, April 2006.

[69] M. Grötschel, L. Lovász, and A. Schrijver, "The ellipsoid method and its consequences in combinatorial optimization." *Combinatorica*, vol. 1, no. 2, pp. 169–197, 1981.

[70] ——, *Geometric Algorithms and Combinatorial Optimization*. Springer Verlag, 1988, 2nd edition in 1993.

[71] A. Schrijver, "A combinatorial algorithm minimizing submodular functions in strongly polynomial time," *Combinatorial Theory, Ser. B*, vol. 80, no. 2, pp. 346–355, 2000.

[72] S. Iwata, L. Fleischer, and S. Fujishige, "A combinatorial strongly polynomial-time algorithm for minimizing submodular functions," *J. Assoc. Comput. Mach.*, vol. 48, pp. 761–777, 2001.

[73] B. A. Davey and H. A. Priestley, *Introduction to Lattices and Order*. Cambridge University Press, Cambridge, 1990.

[74] T. Meltzer, C. Yanover, and Y. Weiss, "Globally optimal solutions for energy minimization in stereo vision using reweighted belief propagation," in *Int. Conf. on Computer Vision (ICCV)*, June 2005, pp. 428–435.

[75] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwala, M. Tappen, and C. Rother, "A comparative study of energy minimization methods for Markov random fields," in *European Conf. Computer Vision (ECCV)*, 2006, pp. II: 16–29.

[76] M. P. Kumar, P. H. S. Torr, and A. Zisserman, "Solving Markov random fields using second order cone programming," in *Conf. on Computer Vision and Pattern Recognition (CVPR)*, 2006.

[77] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 23, no. 11, pp. 1222–1239, 2001.

[78] N. Komodakis and G. Tziritas, "A new framework for approximate labeling via graph cuts." in *Intl. Conf. Computer Vision (ICCV)*, 2005, pp. 1018–1025.

[79] V. Kolmogorov and C. Rother, "Comparison of energy minimization algorithms for highly connected graphs," in *European Conf. Computer Vision (ECCV)*, 2006, pp. II: 1–15.

[80] C. Yanover, T. Meltzer, and Y. Weiss, "Linear programming relaxations and belief propagation: An empirical study," *Machine Learning Research*, vol. 7, pp. 1887–1907, September 2006.

Revisiting the Linear Programming Relaxation Approach to Gibbs Energy Minimization and Weighted Constraint Satisfaction

Tomáš Werner

Abstract—We present a number of contributions to the LP relaxation approach to weighted constraint satisfaction (= Gibbs energy minimization). We link this approach to many works from constraint programming, which relation has so far been ignored in machine vision and learning. While the approach has been mostly considered only for binary constraints, we generalize it to n-ary constraints in a simple and natural way. This includes a simple algorithm to minimize the LP-based upper bound, n-ary max-sum diffusion – however, we consider using other bound-optimizing algorithms as well. The diffusion iteration is tractable for a certain class of high-arity constraints represented as a black-box, which is analogical to propagators for global constraints CSP. Diffusion exactly solves permuted n-ary supermodular problems. A hierarchy of gradually tighter LP relaxations is obtained simply by adding various zero constraints and coupling them in various ways to existing constraints. Zero constraints can be added incrementally, which leads to a cutting plane algorithm. The separation problem is formulated as finding an unsatisfiable subproblem of a CSP.

Index Terms—weighted constraint satisfaction, Gibbs distribution, graphical model, Markov random field, linear programming relaxation, marginal polytope, cut polytope, cutting plane algorithm, global constraint, supermodularity, tree-reweighted max-product

1 Introduction

The topic of this paper is the following problem: given a set of discrete variables and a set of functions each depending on a subset of the variables, maximize the sum of the functions over all the variables. This NP-hard combinatorial optimization problem is known as the weighted (valued, soft) constraint satisfaction problem (WCSP) [1], minimizing Gibbs energy, or finding the most probable configuration of a Markov random field. For Boolean (= two-state) variables, it becomes *pseudo-Boolean optimization* [2]. The WCSP is useful in many fields, such as AI or machine vision and learning.

One of the approaches to WCSP is the linear programming (LP) relaxation, first proposed by Schlesinger [3]. The WCSP is formulated as an integer LP in which the integrality constraint is then relaxed. The dual of the resulting LP minimizes an upper bound on the WCSP optimum by equivalent transformations (reparameterizations) of the problem. Schlesinger and colleagues proposed two algorithms to decrease the bound: *max-sum diffusion* [4], [5], which averages overlapping edge max-marginals until they all coincide, and the *augmenting DAG algorithm* [6]. In general, these algorithms do not find the global minimum of the bound but only a (good) local optimum. We surveyed works [3], [4], [6] in [7], [8].

This article is a continuation of our survey [8] of the approach by Schlesinger et al. and an improved version of our paper [9]. We present the following contributions:

Links to constraint programming: Minimizing Gibbs energy and WCSP are closely linked to the constraint satisfaction problem (CSP) and the related field of constraint programming [10] because: (i) the WCSP upper bound is tight iff the CSP formed by active constraint values has a solution [8], (ii) CSP is a special case of WCSP, (iii) WCSP itself is subject to research in the constraints community [1], [11], [12]. Though early seminal works on using constraints in image analysis reflected the rôle of crisp constraints [13] and their relation to soft constraints [14], [3], nowadays the relation to CSP is ignored in machine vision and learning, where people speak only about MRFs and graphical models. We relate the LP relaxation approach to many results from constraint programming. This links MRF inference to a lot of relevant literature.

N-ary generalization of the LP relaxation: The LP relaxation by Schlesinger and max-sum diffusion were originally formulated for binary WCSPs [3], [8]. We generalize them to constraints of any arity: while in the binary case nodes are coupled to edges, here we couple pairs of hyperedges. Which hyperedge pairs are actually coupled is specified by the *coupling scheme*. This allows to include non-binary constraints in a native way (= not by translation to binary constraints).

High-arity and global constraints: A high-arity constraint represented by a black box is feasible to handle by max-sum diffusion whenever max-marginals of its reparameterization are tractable to compute. This is very similar to how global constraints are commonly treated in CSP.

Supermodular n-ary problems: We show that for supermodular n-ary WCSPs, any local optimum of the bound solves the WCSP exactly; here, it suffices to couple

• *The author is with the Department of Cybernetics, Czech Technical University, Karlovo náměstí 13, 12135 Praha, Czech Republic. Email: werner@cmp.felk.cvut.cz.*

hyperedges only to variables, i.e., to achieve only generalized arc consistency. By revisiting [15], [12], we generalize this result to permuted supermodular WCSPs.

Tighter relaxations: We show that once we have a natively n-ary LP relaxation, it can be tightened simply by adding zero constraints. Dually, this means that equivalent transformations change not only the constraint values but also the problem hypergraph. Adding various zero constraints and coupling them to existing constraints in various ways yields a hierarchy of gradually tighter relaxations, corresponding to a hierarchy of nested polyhedral outer bounds of the marginal polytope. This can be done incrementally, yielding a dual cutting plane algorithm. The separation problem can be posed purely in CSP terms, such as finding an unsatisfiable sub-CSP. We relate higher-order LP relaxations to stronger local consistencies (path consistency, k-consistency) in CSP.

1.1 Other Works

Non-binary constraints, tighter relaxations and cutting-plane strategies in LP relaxation approaches to WCSP have been addressed in a number of other works.

Most similar to ours is the *decomposition* approach. The original WCSP is expressed as a sum of subproblems on tractable hypergraphs. The sum of maxima of the subproblems is an upper bound on the maximum of their sum (= the true solution). The bound is minimized over constraint values of the subproblems, subject to that they sum to a reparameterization of the original WCSP.

The approach was proposed by Wainwright et al. [16], [17], [18] for tree-structured subproblems and improved by Kolmogorov [19]. Using hypertrees rather than trees allows for natively handling non-binary constraints and yields a hierarchy of progressively tighter relaxations [17, §VI], [18]. Johnson et al. [20] used general subproblems rather than (hyper)trees, also obtaining a hierarchy of relaxations. Komodakis et al. [21] pointed out that decomposition is a standard technique in optimization [22].

Our approach can be seen equivalent to the decomposition approach. In the one direction, we decompose the WCSP into the smallest possible subproblems, individual constraints. In the other direction, if each constraint in our approach is *itself* defined as a sum of several constraints, we obtain the decomposition approach. Our adding of zero constraints is similar to constructing an *augmented hypergraph* in [20], [18].

Weiss et al. [23] extended the LP relaxation to n-ary problems in a way similar to ours, with a small but crucial difference: they couple hyperedges only to variables (rather than other hyperedges), in which case adding zero constraints does not tighten the relaxation.

A global constraint in WCSP was used by Rother et al. [24]. Our aproach is different, relying on LP relaxation.

Tighter LP relaxations of WCSP and cutting plane strategies have recently appeared in many works. These approaches can be divided into primal [25], [26] and dual [20], [27], [28], [29]. Ours is dual. Primal approaches have a drawback that no algorithms to solve the primal LP are known that scale to large problems.

Koster et al. [25] proposed the same LP relaxation as [3] (without dual) and a primal cutting plane algorithm.

Sontag and Jaakkola [26] observed the relation of the marginal polytope and the cut polytope [30], for which many classes of cutting planes are known, and adapted the algorithm [31] to separate inconsistent cycles.

Kumar and Torr [27] and Komodakis and Paragios [28] add cycles to the LP relaxation.

Sontag et al. [29] incrementally tighten the relaxation by adding clusters of variables.

N-ary generalizations of the LP relaxation to WCSP, their higher-order versions, and cutting plane strategies proposed in the above works often use different formalisms which makes it difficult to compare them – however, one can conjecture that they all yield the same hierarchy of polyhedral relaxations (or at least some of its lower levels). We offer yet another formulation, which is very simple and general. Its main strength is in its close relation to constraint programming, which allows to formulate optimality conditions and the separation problem in CSP terms and straightforwardly extends to global and n-ary (permuted) supermodular constraints.

2 Notation and Problem Formulation

2^V resp. $\binom{V}{k}$ is the set of all resp. of k-element subsets of a set V. The value $[\![\omega]\!]$ is 1 if logical expression ω is true and 0 if ω is false. $\mathbb{R}$ denotes the reals and $\bar{\mathbb{R}} = \mathbb{R} \cup \{-\infty\}$. The set of mappings from X to Y is Y^X. An ordered resp. unordered tuple is denoted $(\cdots)$ resp. $\{\cdots\}$.

Let V be a finite, totally ordered set of **variables**. To emphasize the variable ordering, when defining a subset of V by enumerating its elements we use $(\cdots)$ instead of $\{\cdots\}$. Each variable $v \in V$ is assigned a finite set X_v, its **domain**. An element of X_v is a **state** of variable v and is denoted x_v. The **joint domain** of variables $A \subseteq V$ is the Cartesian product $X_A = \bigtimes_{v \in A} X_v$, where the order of the factors is determined by the order on V. A tuple $x_A \in X_A$ is a **joint state** of variables A.

Example 1. Let $V = (1, 2, 3, 4)$, $X_1 = X_2 = X_3 = X_4 = \{\mathsf{a}, \mathsf{b}\}$. A joint state $x_{134} = (x_1, x_3, x_4) \in X_{134} = X_1 \times X_3 \times X_4$ of variables $A = (1, 3, 4) \subseteq V$ is e.g. $(\mathsf{a}, \mathsf{a}, \mathsf{b})$. □

We will use the following **implicit restriction convention**: for $B \subseteq A$, whenever symbols x_A and x_B appear in a single expression they do not denote independent joint states but x_B denotes the restriction of x_A to variables B.

A **constraint** with **scope** $A \subseteq V$ is a function $f_A \colon X_A \to \bar{\mathbb{R}}$. The **arity** of the constraint is the size of its scope, $|A|$.

Let $E \subseteq 2^V$ be a set of subsets of V, i.e., a hypergraph. Each hyperedge $A \in E$ is assigned a constraint f_A. All these constraints together are understood as a single mapping $f \colon T(E, X_V) \to \bar{\mathbb{R}}$, $(A, x_A) \mapsto f_A(x_A)$, where we denoted $T(E, X_V) = \{\, (A, x_A) \mid A \in E,\ x_A \in X_A \,\}$.

The topic of this article is the problem

$$\max_{x_V \in X_V} \sum_{A \in E} f_A(x_A) \tag{1}$$

$$\sum_{A\in E}\sum_{x_A} f_A(x_A)\,\mu_A(x_A) \to \max_{\mu} \qquad \sum_{A\in E}\psi_A \to \min_{\varphi,\psi} \tag{2a}$$

$$\sum_{x_{A\setminus B}} \mu_A(x_A) = \mu_B(x_B) \qquad \varphi_{A,B}(x_B) \leqslant 0 \qquad \forall (A,B)\in J,\ x_B\in X_B \tag{2b}$$

$$\sum_{x_A} \mu_A(x_A) = 1 \qquad \psi_A \leqslant 0 \qquad \forall A\in E \tag{2c}$$

$$\mu_A(x_A) \geq 0 \qquad \sum_{B|(B,A)\in J}\varphi_{A,B}(x_A) - \sum_{B|(A,B)\in J}\varphi_{B,A}(x_B) + \psi_A \geq f_A(x_A) \quad \forall A\in E,\ x_A\in X_A \tag{2d}$$

which we will refer to as the **weighted constraint satisfaction problem** (WCSP). The WCSP instance is defined by a tuple (V, E, X_V, f). When V, E and X_V are clear from context, sometimes we will refer to the instance just as f. The **arity of the instance** is $\max_{A\in E}|A|$.

Example 2. Let $V = (1,2,3,4)$ and $E = \{(2,3,4), (1,2), (3,4), (3)\}$. Problem (1) means that we maximize the function $f_{234}(x_2,x_3,x_4)+f_{12}(x_1,x_2)+f_{34}(x_3,x_4)+f_3(x_3)$ over x_1, x_2, x_3, x_4. □

3 Linear Programming Relaxation

The LP relaxation approach developed by Schlesinger [3] was originally formulated for binary WCSP. Following our survey [8], we generalize it here to n-ary WCSPs.

We start by writing the relaxation as the pair of mutually dual linear programs (2). Here, $\leqslant 0$ means that the variable is unconstrained. In matrix form, (2) reads

$$f^\top\mu \to \max_{\mu} \qquad \psi^\top 1 \to \min_{\varphi,\psi} \tag{3a}$$

$$M\mu = 0 \qquad \varphi \leqslant 0 \tag{3b}$$

$$N\mu = 1 \qquad \psi \leqslant 0 \tag{3c}$$

$$\mu \geq 0 \qquad \varphi^\top M + \psi^\top N \geq f^\top \tag{3d}$$

The pairs (2) and (3) are written such that a constraint and its Lagrange multiplier is always on the same line.

Besides E, X_V and f, the LP is described by a set

$$J \subseteq I(E) = \{\,(A,B) \mid A\in E,\ B\in E,\ B\subset A\,\} \tag{4}$$

where $I(E)$ denotes the (strict) inclusion relation on E. We refer to the set J as the **coupling scheme**.

In §3.1, §3.2 we explain the primal and dual in detail.

3.1 The Primal Program

In the primal LP, each hyperedge $A\in E$ is assigned a function $\mu_A\colon X_A \to [0,1]$, where primal constraints (2c)+(2d) impose that μ_A is a probability distribution. All the distributions together are understood as a mapping $\mu\colon T(E,X_V) \to [0,1]$. While the constraints (2c)+(2d) affect each distribution separately, constraint (2b) couples some pairs of distributions, imposing that they have consistent marginals. The coupling scheme J determines which pairs of distributions are actually coupled.

Example 3. Let $A=(1,2,3)$, $B=(2,3)$. Then (2b) reads: $\forall x_2\in X_2, x_3\in X_3\colon \sum_{x_1}\mu_{123}(x_1,x_2,x_3) = \mu_{23}(x_2,x_3)$. □

If μ is integral, $\mu\colon T(E,X_V)\to\{0,1\}$, then, on certain conditions on E and J, the primal LP is equivalent to WCSP. Theorem 1 shows that for this equivalence to hold, it suffices to couple hyperedges to variables [23].

Theorem 1. *Let* $\binom{V}{1}\subseteq E$ *and* $I_{\mathrm{GAC}}(E)\subseteq J$, *where*

$$I_{\mathrm{GAC}}(E) = \{\,(A,(v)) \mid A\in E,\ |A|>1,\ v\in A\,\} \tag{5}$$

Then the primal LP with integral μ *is equivalent to (1).*

Proof: Let μ be integral. Then μ_A represents a single joint state, x_A. Thus, $f_A(x_A) = \sum_{y_A} f_A(y_A)\mu_A(y_A)$. Equality (2b) means that the joint state represented by μ_B is the *restriction* of the joint state represented by μ_A on variables B. If $I_{\mathrm{GAC}}(E)\subseteq J$ then fixing μ_v for all $v\in V$ uniquely determines μ_A for all $A\in E$. □

If $I_{\mathrm{GAC}}(E)\not\subseteq J$ then the primal LP can have integral optimal solutions that are not solutions of (1).

To conclude, the primal LP is a relaxation of the WCSP. The relaxation is twofold: first, μ is allowed to be continuous rather than integral, second, only a subset, J, of possible marginalization constraints is imposed. Clearly, the primal optimum is an *upper bound* on (1).

3.1.1 Alternative Forms of Marginal Consistency

The marginal consistency condition (and the coupling scheme) could be formulated in several alternative ways, different from (2b). We state these alternative forms here.

First, (2b) can be stated in a symmetric form as

$$\sum_{x_{A\setminus C}}\mu_A(x_A) = \sum_{x_{B\setminus C}}\mu_B(x_B) \qquad \begin{cases}\forall (A,B,C)\in J\\ \forall x_C\in X_C\end{cases} \tag{6}$$

where $J\subseteq I(E) = \{\,(A,B,C) \mid A,B\in E,\ \emptyset\neq C\subseteq A\cap B\,\}$. Form (6) may appear more general than the asymmetric form (2b); e.g., if $A\cap B\neq\emptyset$ and $B\not\subset A$ then equality (2b) is vacuous and (6) is not. But this is not so because equality (6) applied on (A,B,C) is equivalent to two equalities (2b) applied on (A,C) and (B,C). This assumes that $C\in E$, which can be ensured by adding the zero constraint with scope C (see Example 6 in §9).

Second, while equality (2b) is imposed on all joint states $x_B\in X_B$, we could impose it only on their subset:

$$\sum_{x_{A\setminus B}}\mu_A(x_A) = \mu_B(x_B) \qquad \forall (A,B,x_B)\in J \tag{7}$$

where $J \subseteq I(E) = \{(A, B, x_B) \mid A, B \in E,\ B \subset A,\ x_B \in X_B\}$. Form (6) can be refined similarly.

3.2 The Dual Program

Definition 1. *Let $E \subseteq 2^V$, $E' \subseteq 2^V$, $f\colon T(E, X_V) \to \bar{\mathbb{R}}$, $f'\colon T(E', X_V) \to \bar{\mathbb{R}}$. WCSP instances (V, E, X_V, f) and (V, E', X_V, f') are* **equivalent** *iff*

$$\sum_{A \in E} f_A(x_A) = \sum_{A \in E'} f'_A(x_A) \qquad \forall x_V \in X_V$$

Unlike the previous definition of WCSP equivalence [3], [8], [19], Definition 1 does not require that equivalent WCSPs have the same hypergraph, it only requires that they have the same variables V and domains X_V. Thus, it allows to change not only the constraint values but also the hypergraph. We call a transformation taking a WCSP to its equivalent an **equivalent transformation**. Until §9, we will consider only equivalent transformations that preserve the hypergraph (i.e., $E = E'$ in the definition).

The simplest hypergraph-preserving equivalent transformation is applied to a single pair of constraints, f_A and f_B for $B \subset A$, by adding a function $\varphi_{A,B}\colon X_B \to \mathbb{R}$ (a 'message') to f_A and subtracting it from[1] f_B, i.e.,

$$f_A(x_A) \leftarrow f_A(x_A) + \varphi_{A,B}(x_B) \qquad \forall x_B \in X_B \qquad (8a)$$
$$f_B(x_B) \leftarrow f_B(x_B) - \varphi_{A,B}(x_B) \qquad \forall x_B \in X_B \qquad (8b)$$

Let a function $\varphi_{A,B}\colon X_B \to \mathbb{R}$ be assigned to each $(A, B) \in J$. The collection of these functions forms a single mapping φ. Let f^φ denote the WCSP obtained by applying (8) on f for all $(A, B) \in J$, i.e., f^φ is given by

$$f^\varphi_A(x_A) = f_A(x_A) - \sum_{B|(B,A)\in J} \varphi_{B,A}(x_A) + \sum_{B|(A,B)\in J} \varphi_{A,B}(x_B) \qquad (9)$$

We refer to (9) as a **reparameterization**[2] of f.

In matrix form, (9) reads $f^\varphi = f - M^\top \varphi$. This shows clearly why reparameterizations preserve the primal objective: because $M\mu = 0$ implies $(f^\top - \varphi^\top M)\mu = f^\top \mu$.

Theorem 2. *For any $f\colon T(E, X_V) \to \bar{\mathbb{R}}$, we have*

$$\max_{x_V} \sum_{A \in E} f_A(x_A) \le \sum_{A \in E} \max_{x_A} f_A(x_A) \qquad (10)$$

which holds with equality iff there exists a joint state $x_V \in X_V$ such that $f_A(x_A) = \max_{y_A} f_A(y_A)$ for all $A \in E$.

Proof: Clearly, $\max_i \sum_j a_{ij} \le \sum_j \max_i a_{ij}$ for any $a_{ij} \in \bar{\mathbb{R}}$, which holds with equality iff there exists i such that $a_{ij} = \max_k a_{kj}$ for all j. This is applied to (1). □

1. While (8) is clearly an equivalent transformation, it is far from obvious whether *any* hypergraph-preserving equivalent transformation is realizable as a composition of (8) for various $(A, B) \in I(E)$. In analogy with [8, Theorem 3] and [19, Lemma 6.3], we conjecture that this is so if E is closed to intersection ($A, B \in E$ implies $A \cup B \in E$).

2. There is a 'gauge freedom' in (9): $f = f^\varphi$ need not imply $\varphi = 0$. It is an open problem for a given f to describe the set $\{\varphi \mid f = f^\varphi\}$. If some constraint values are $-\infty$, this seems to be difficult.

The right-hand expression in (10) is an upper bound on (1). By eliminating variables ψ_A, the dual LP reads

$$\min_{\varphi} \sum_{A \in E} \max_{x_A} f^\varphi_A(x_A) \qquad (11)$$

which can be interpreted as *minimizing the upper bound by reparameterizations* permitted by J.

3.3 Hierarchy of LP Relaxations

We have shown, both by primal and dual arguments, that the optimum of the LP (2) is an upper bound on the true WCSP optimum (1). Sometimes, the bound is tight, i.e., equal to (1). For any non-trivially chosen J, this happens for a large and complex class of WCSPs.

Tightness of the relaxation depends on the coupling scheme J. An equality (2b) in the primal corresponds via duality to a variable $\varphi_{A,B}(x_B)$ in the dual – thus, the larger J is, the more the primal is constrained and the larger is the set of permitted reparameterizations in the dual. LP relaxations for various $J \in I(E)$ form a *hierarchy*, partially ordered by the inclusion on $I(E)$.

Let $P(E, X_V, J) \subseteq [0, 1]^{T(E,X_V)}$ denote the polytope of mappings μ feasible to the primal LP. The hierarchy of relaxations is established by the obvious implication[3]

$$J_1 \supseteq J_2 \implies P(E, X_V, J_1) \subseteq P(E, X_V, J_2) \qquad (12)$$

Imposing marginal consistency in form (7) rather than (2b) would yield a finer-grained hierarchy of relaxations. This would require to modify formula (9).

4 Constraint Satisfaction Problem

The *constraint satisfaction problem* (CSP) [33] is one of the classical NP-complete problems. Here we give background on the CSP which we will need later.

Let each hyperedge $A \in E$ be assigned a crisp constraint $\bar{f}_A\colon X_A \to \{0, 1\}$, understood as the characteristic function of an $|A|$-ary relation over variables A. A joint state x_A is **permitted** (**forbidden**) iff $A \in E$ and $\bar{f}_A(x_A)$ equals 1 (0). Let $\vee$ ($\wedge$) denote the logical disjunction (conjunction). The CSP asks whether there exists a joint state $x_V \in X_V$ satisfying all the relations, i.e., $\bar{f}_A(x_A) = 1$ for each $A \in E$. Such x_V is a **solution**. The CSP instance is defined by $(V, E, X_V, \bar{f})$, where $\bar{f}\colon T(E, X_V) \to \{0, 1\}$.

A CSP is **satisfiable** iff it possesses a solution. We call x_A a **satisfiable joint state** iff it can be extended to a solution. Note, the fact that the CSP is satisfiable and x_A is permitted does not imply that x_A is satisfiable. A joint state x_A is **locally consistent** iff $\bar{f}_B(x_B) = 1$ for every B such that $B \in E$ and $B \subseteq A$. In particular, x_V is a solution iff it is locally consistent.

For tractable subclasses of the CSP see [34], [35].

A powerful tool to solve CSPs is *constraint propagation* [36] (*filtering*, *relaxation labeling* [14]). The possibility to

3. Different coupling schemes may yield the same relaxation, i.e., $P(E, X_V, J_1) = P(E, X_V, J_2)$ need not imply $J_1 = J_2$. It is an open problem to characterize when exactly this happens.

propagate constraints is the distinguishing feature of the CSP: while in general a search cannot be avoided to solve a CSP, propagating constraints during the search prunes the search space such that instances of practical size can be solved. In a way, constraint propagation is a crisp analogy of 'message passing' in graphical models.

In constraint propagation, some obviously unsatisfiable joint states are iteratively deleted using a simple local rule, a *propagator*. This is often done until the CSP satisfies a state characterized by a *local consistency* – then we speak about *enforcing the local consistency*.

Many local consistencies have been proposed, see [36] for a survey and [37] for comparison of their strength for binary CSPs. The most well-known one is *arc consistency* (AC). It is defined for binary CSPs, while we need a local consistency defined for CSPs of any arity. Many such consistencies are known; of them, most relevant to our LP relaxation are *pairwise consistency* (PWC), *generalized arc consistency* (GAC), and *k-consistency* [36].

4.1 J-consistency

To fit our form of coupling, we introduce a modification of PWC, *J-consistency*. While PWC enforces consistency of all pairs of relations, J-consistency enforces consistency of relations $\bar{f}_A$ and $\bar{f}_B$ only if $(A,B) \in J$.

Definition 2. *For $B \subset A$, relations $\bar{f}_A\colon X_A \to \{0,1\}$ and $\bar{f}_B\colon X_B \to \{0,1\}$ are* **pairwise consistent** *iff*

$$\bigvee_{x_{A\setminus B}} \bar{f}_A(x_A) = \bar{f}_B(x_B) \qquad \forall x_B \in X_B \tag{13}$$

A CSP $(V,E,X_V,\bar{f})$ is **J-consistent** *iff relations $\bar{f}_A$ and $\bar{f}_B$ are pairwise consistent for every $(A,B) \in J$.*

Note that the set of equalities (13) has the following meaning: a joint state x_B is permitted by relation $\bar{f}_B$ iff x_B can be extended to a joint state x_A satisfying $\bar{f}_A$.

PWC and GAC are special cases of J-consistency. PWC is obtained if E is closed to intersection (i.e., $A,B \in E$ implies $A \cap B \in E$) and $J = I(E)$. GAC is obtained[4] if $\binom{V}{1} \subseteq E$ and $J = I_{\mathrm{GAC}}(E)$. For binary CSPs with $\binom{V}{1} \subseteq E$, PWC and GAC become AC.

To enforce J-consistency, a generalization of well-known algorithms to enforce (G)AC can be used. Algorithm 1 deletes unsatisfiable joint states until the CSP becomes J-consistent, while preserving the solution set, i.e., the relation $\bigwedge_{A\in E} \bar{f}_A(x_A)$.

Obviously, if the algorithm makes $\bar{f}$ empty (i.e., $\bar{f} = 0$) then the initial CSP was unsatisfiable. Note that if any relation $\bar{f}_A$ becomes empty during the algorithm, it is already clear that $\bar{f}$ will eventually become empty.

We give the algorithm also in the parameterized form as Algorithm 2, which does not change the relations $\bar{f}$ (thus, they can be represented intensionally, §6.3). Each

4. We remark that the concept of GAC allows us to explain Theorem 1 in §3.1 in CSP terms as follows. An integer primal-feasible μ can be seen as a CSP in which, due to (2c), each relation has a single permitted joint state. Clearly, such a CSP is satisfiable iff it is GAC.

Algorithm 1 (enforcing J-consistency of CSP)

repeat
  Find $(A,B) \in J$, $x_B \in X_B$ s.t. $\bigvee_{x_{A\setminus B}} \bar{f}_A(x_A) \neq \bar{f}_B(x_B)$
  for $x_{A\setminus B} \in X_{A\setminus B}$ **do** $\bar{f}_A(x_A) \leftarrow 0$ **end for**
  $\bar{f}_B(x_B) \leftarrow 0$
until $\bar{f}$ is J-consistent

$(A,B) \in J$ is assigned a function $\bar{\varphi}_{A,B}\colon X_B \to \{0,1\}$, where all these functions together form a mapping $\bar{\varphi}$. Initially we set $\bar{\varphi} = 1$. Analogically to (9), we define transformation $\bar{f}^{\bar{\varphi}}$ of $\bar{f}$ by

$$\bar{f}^{\bar{\varphi}}_A(x_A) = \bar{f}_A(x_A) \wedge \bigwedge_{B|(B,A)\in J} \bar{\varphi}_{B,A}(x_A) \;\wedge \bigwedge_{B|(A,B)\in J} \bar{\varphi}_{A,B}(x_B)$$

Algorithm 2 (enforcing J-consistency, parameterized)

repeat
  Find $(A,B) \in J$, $x_B \in X_B$ s.t. $\bigvee_{x_{A\setminus B}} \bar{f}^{\bar{\varphi}}_A(x_A) \neq \bar{f}^{\bar{\varphi}}_B(x_B)$
  $\bar{\varphi}_{A,B}(x_B) \leftarrow 0$
until $\bar{f}^{\bar{\varphi}}$ is J-consistent

The *closure* of a CSP with respect to a local consistency is the maximal subset of its permitted joint states that still achieves the local consistency [36, §3]. To formalize this, we define **inclusion** $\leq$ and **join** $\vee$ on CSPs by:

$$\bar{f} \leq \bar{f}' \quad \iff \forall A, x_A\colon \bar{f}_A(x_A) \leq \bar{f}'_A(x_A)$$
$$\bar{f} = \bar{f}' \vee \bar{f}'' \iff \forall A, x_A\colon \bar{f}_A(x_A) = \bar{f}'_A(x_A) \vee \bar{f}''_A(x_A)$$

Definition 3. *The* **J-consistency closure** *of a CSP $(V,E,X_V,\bar{f})$ is the CSP $(V,E,X_V,\bar{f}^*)$ where*

$$\bar{f}^* = \bigvee \{\, \bar{f}' \mid \bar{f}' \leq \bar{f},\ \ \bar{f}' \text{ is } J\text{-consistent} \,\} \tag{14}$$

It is easy to verify that the join of J-consistent CSPs is J-consistent (in other words, J-consistent CSPs form a join-semilattice). Hence the closure (14) is J-consistent.

It is not true in general that an algorithm to enforce a local consistency produces the closure of that local consistency [36]. However, it is true for J-consistency.

Theorem 3. *Algorithm 1 or 2 finds the J-consistency closure.*

4.2 k-consistency

There exist stronger local consistencies than (G)AC and PWC. Most important of them is *k-consistency* [36].

Definition 4. *A CSP is* **k-consistent** *iff for every locally consistent joint state x_A such that $|A| = k-1$ and every variable $v \in V$ there exists a state x_v such that $x_{A\cup(v)}$ is locally consistent (i.e., x_A can be extended to variable v).*

Strong k-consistency is k'-consistency for all $k' \leq k$. Solvability by strong k-consistency characterizes an important class of tractable relation languages [34, §8.4.2] (e.g., binary CSPs with Boolean variables are solved by

strong 3-consistency). Strong k-consistency solves CSPs with (hyper)graph of treewidth less than k [38], [35, §3.4].

Unlike J-consistency, enforcing (strong) k-consistency in general requires adding new relations to the CSP. Let $\bar{f}_A = 1$ denote the *universal relation* (i.e., identically true) on A. An inefficient way to enforce k-consistency is to add all possible universal relations of arity $k-1$ and k (such that $\binom{V}{k-1} \cup \binom{V}{k} \subseteq E$), then enforce PWC, and then remove all the previously added k-ary relations [39, §8].

In a more efficient algorithm, only *some* of the missing $(k-1)$-ary relations can be added. It achieves strong k-consistency by enforcing k'-consistency for $k' = 2, \dots, k$ in turn. A $(k-1)$-consistent CSP is made k-consistent as follows. We iteratively set $\bar{f}_A(x_A) \leftarrow 0$ whenever $|A| = k-1$ and x_A cannot be extended to some variable v. Here, if A was not already in E, we first add A to E and set $\bar{f}_A \leftarrow 1$. Simultaneously, PWC is enforced.

2-consistency is the same as arc consistency.

4.2.1 3-consistency and Path Consistency

3-consistency in a binary CSP is also known as *path consistency* for the following reason. A sequence $(u, \dots, v)$ of variables is a *path* if $\{u, v\}$ and all edges along the sequence are in E (we also allow $u = v$ which yields a cycle). The path is *consistent* iff any state pair (x_u, x_u) satisfying relation $\bar{f}_{uv}$ can be extended to all intermediate relations along the path. A graph is chordal (= triangulated) iff every cycle of length 4 or more has a chord.

Theorem 4. *In a chordal graph, every path of length 3 (i.e., with 3 variables) is consistent iff every path is consistent.*

Proof: For complete (hence chordal) graphs, this is a classical results by Montanari [13], [36]. It was extended to chordal graphs by Bliek and Sam-Haroud [40]. □

By definition, 3-consistency means that any locally consistent state pair (x_u, x_v) can be extended to any third variable w. In other words, after filling-in the CSP to the complete graph with universal binary relations, all paths of length 3 are consistent – hence, all paths are consistent.

4.3 CSP with a Relation over All the Variables

Let $\bigcup E = \bigcup_{A \in E} A$ (typically but not necessarily we have $\bigcup E = V$). Consider a CSP containing a relation over hyperedge $\bigcup E$ (i.e., $\bigcup E \in E$) and the coupling scheme

$$I_{\mathrm{SAT}}(E) = \{\, (\textstyle\bigcup E, A) \mid A \in E,\ A \neq \bigcup E \,\} \tag{15}$$

which couples $\bigcup E$ to all other hyperedges. Propositions 5, 6, 7 give properties of $I_{\mathrm{SAT}}(E)$-consistency we will need later. Proofs are easy, from Definitions 2, 3.

Proposition 5. *A CSP with $\bigcup E \in E$ has a non-empty $I_{\mathrm{SAT}}(E)$-consistency closure iff it is satisfiable.*

Proposition 6. *A CSP with $\bigcup E \in E$ is $I_{\mathrm{SAT}}(E)$-consistent iff every joint state x_A permitted by $\bar{f}_A$ is satisfiable.*

Proposition 7. *If a CSP with $\bigcup E = V \in E$ is $I_{\mathrm{SAT}}(E)$-consistent then*

$$\bar{f}_V(x_V) \leq \bigwedge_{A \in E} \bar{f}_A(x_A) \qquad \forall x_V \in X_V \tag{16}$$

Note, equality in (16) for all x_V means that the relation $\bar{f}_V$ is realizable as the conjunction of the relations $\bar{f}_A$.

Example 4. Let $V = (1,2,3)$, $X_V = \{0,1\}^V$, and $E = \{\,(1,2),(1,3),(2,3),(1,2,3)\,\}$. Let $\bar{f}$ be defined by

$$\bar{f}_{123}(x_1,x_2,x_3) = x_1\bar{x}_2\bar{x}_3 \vee \bar{x}_1x_2\bar{x}_3 \vee \bar{x}_1\bar{x}_2x_3 \tag{17a}$$

$$\bar{f}_{12}(x_1,x_2) = \bar{x}_1\bar{x}_2 \vee x_1\bar{x}_2 \vee \bar{x}_1x_2 \tag{17b}$$

and $\bar{f}_{13} = \bar{f}_{23} = \bar{f}_{12}$, where we denoted $\bar{x}_u = 1 - x_u$ and $x_ux_v = x_u \wedge x_v$. The CSP $(V, E, X_V, \bar{f})$ is $I_{\mathrm{SAT}}(E)$-consistent but the inequality in (16) is strict for $x_V = (x_1,x_2,x_3) = (0,0,0)$. Clearly, $\bar{f}_{123}$ is not realizable as a conjunction of *any* binary relations. □

5 Optimality of the LP Relaxation

Given feasible primal and dual variables, we want to recognize whether they are optimal to the LP pair and whether this optimum is tight. As shown in [3], [8] for binary WCSPs, the answers to these questions depend only on the properties of the *CSP formed by the active joint states*. This is significant because it moves reasoning about optimality of the LP relaxation to the realm of a well-known and long-studied problem.

Here we extend these results for WCSPs of any arity. Theorems 8, 9, 10 characterize three levels of optimality of the LP relaxation: whether the upper bound is tight (i.e., equal to (1)), minimal, or locally minimal.

Definition 5. *Given a function $f_A\colon X_A \to \bar{\mathbb{R}}$, we define a relation $\lceil f_A \rceil\colon X_A \to \{0,1\}$ by*

$$\lceil f_A \rceil(x_A) = \begin{cases} 1 & \text{if } f_A(x_A) = \max\limits_{y_A} f_A(y_A) \\ 0 & \text{if } f_A(x_A) < \max\limits_{y_A} f_A(y_A) \end{cases} \tag{18}$$

A joint state x_A of constraint f_A is **active** *iff $\lceil f_A \rceil(x_A) = 1$.*

Given a mapping $f\colon T(E, X_V) \to \bar{\mathbb{R}}$, we define a mapping $\lceil f \rceil\colon T(E, X_V) \to \{0,1\}$ by $\lceil f \rceil_A(x_A) = \lceil f_A \rceil(x_A)$.

Theorem 8. *Inequality (10) holds with equality iff the CSP $(V, E, X_V, \lceil f \rceil)$ is satisfiable*[5]*. The solutions of this CSP are in one-to-one correspondence with the maximizers of (1).*

Proof: By restating the second part of Theorem 2. □

Theorem 9. *Let $\mu\colon T(E, X_V) \to [0,1]$ be feasible to the primal LP. The primal and dual LP are jointly optimal iff*

$$[1 - \lceil f_A^{\varphi} \rceil(x_A)]\, \mu_A(x_A) = 0 \qquad \forall A \in E,\ x_A \in X_A \tag{19}$$

Proof: Apply complementary slackness to (2b). □

Theorem 9 characterizes WCSPs for which the bound is dual optimal, i.e., cannot be improved by changing φ:

5. Note a subtlety: since finding a solution to a CSP is NP-complete even if we know that the CSP is satisfiable, finding an optimizer to a WCSP is NP-complete even if we know that the upper bound is tight.

it is when a primal feasible μ exists such that $\mu_A(x_A) = 0$ whenever joint state x_A is inactive. In fact, given any single dual optimal solution, all primal optimal solutions are uniquely determined by the active joint states by (19).

Theorem 10. *If, for any J, φ is optimal to the dual LP then the J-consistency closure of $\lceil f^\varphi \rceil$ is not empty.*

Proof: Let $\bar{f}'_A(x_A) = [\![\mu_A(x_A) > 0]\!]$. For any non-negative μ_A, obviously $\sum_{x_{A\setminus B}} \mu_A(x_A) = \mu_B(x_B)$ implies $\bigvee_{x_{A\setminus B}} \bar{f}'_A(x_A) = \bar{f}'_B(x_B)$. Hence, the CSP $\bar{f}'$ is J-consistent. Clearly, (19) can be rewritten as $\bar{f}' \leq \lceil f^\varphi \rceil$. By (14), $\lceil f^\varphi \rceil$ has a non-empty J-consistency closure. □

As shown in [41], [8], [19], a non-empty closure of $\lceil f^\varphi \rceil$ is only necessary but not sufficient for dual optimality. Thus, Theorem 10 characterizes *local minima* of the upper bound (10). These local minima naturally appear in several algorithms to solve the dual LP[6].

The only known WCSP classes for which all local minima are global are the supermodular ones and those with binary constraints and Boolean variables [8], [42].

6 Optimizing the Bound

Here we focus on algorithms to solve the LP (2).

It is better to solve the dual LP than the primal LP. This is because no algorithm is known to solve (or find a good suboptimum of) the primal for large instances; only general LP solvers (simplex) have been used [25], [26]. Moreover, the number of primal variables is exponential in the arity of the instance, thus for large arities the primal cannot be solved explicitly at all, whereas the corresponding exponential number of dual constraints sometimes can be handled implicitly (§6.3, §9.3).

6.1 Existing Algorithms for Binary Problems

The dual LP in the form (11) is an unconstrained minimization of a convex piecewise linear (hence nonsmooth) function. To scale to large instances, it is reasonable to require that an algorithm to solve (11) have space complexity *linear in the number of dual variables $\varphi_{A,B}(x_B)$*. This rules out e.g. the simplex and interior point algorithms. For binary WCSPs, known algorithms with this property can be divided into two groups:

1) Local algorithms find a local minimum of the upper bound characterized by arc consistency of the active joint states. The found local minima are usually very good or even global. Two types of such algorithms are known:

a) Algorithms based on averaging max-marginals: *max-sum diffusion* [4], [5], [8], *TRW-S* [19] and [43], [20]. They can be roughly seen as a (block) coordinate descent. Existence of local minima follows from the fact that coordinate descent need not find the global minimum of a convex nonsmooth function [44, §7.3].

b) The *augmenting DAG* algorithm [6], [7], [8] and the *virtual arc consistency (VAC)* algorithm [12]. They explicitly try to enforce AC of $\lceil f^\varphi \rceil$. If all the states of any variable are deleted, the bound can be improved by back-tracking the causes of deletions.

2) Global algorithms find the global minimum of the upper bound. Two types of such algorithms are known:

a) Subgradient descent [45], [21] is a well-known method to minimize nonsmooth functions. These approaches rely on decomposing the WCSP as a sum of tractable subproblems (§1.1). To achieve good convergence rate, the subproblems must be well chosen (large enough).

b) Smoothing algorithms [44, §7.4], [23], [20], [46] use a sequence of smooth convex approximations of our nonsmooth convex objective function. Each such function can be minimized by coordinate descent globally.

Unlike the global algorithms, the local algorithms improve the bound monotonically.

In principle, any of the above algorithms can be generalized to n-ary WCSPs, still keeping its space complexity linear in the number of variables $\varphi_{A,B}(x_B)$. This is easiest for max-sum diffusion, which we show in §6.2.

6.2 Max-sum Diffusion

The max-sum diffusion iteration is the reparameterization (8) on a single $(A, B) \in J$ that averages f_B and the max-marginals of f_A, i.e., makes satisfied the equalities

$$\max_{x_{A\setminus B}} f_A(x_A) = f_B(x_B) \qquad \forall x_B \in X_B \tag{20}$$

If $f_B(x_B) > -\infty$, $\max_{x_{A\setminus B}} f_A(x_A) > -\infty$, this is done by setting $\varphi_{A,B}(x_B) = [f_B(x_B) - \max_{x_{A\setminus B}} f_A(x_A)]/2$ in (8).

Theorem 11. *The iteration does not increase the upper bound.*

Proof: Let us denote $a(x_B) = \max_{x_{A\setminus B}} f_A(x_A)$, $b(x_B) = f_B(x_B)$, $c(x_B) = [b(x_B) - a(x_B)]/2 = \varphi_{A,B}(x_B)$. Before the iteration, the contribution of f_A and f_B to the upper bound (10) is

$$\max_{x_A} f_A(x_A) + \max_{x_B} f_B(x_B) = \max_{x_B} a(x_B) + \max_{x_B} b(x_B) \tag{21}$$

After the iteration, this contribution is

$$\begin{aligned} &\max_{x_B}[a(x_B) + c(x_B)] + \max_{x_B}[b(x_B) - c(x_B)] \\ &= \max_{x_B}[a(x_B) + b(x_B)] \end{aligned} \tag{22}$$

Clearly, expression (22) is not greater than (21). □

Using parameterization (9), we obtain Algorithm 3. To correctly handle infinite weights, it assumes that the CSP $\bar{f}^{\text{fin}}$ defined by $\bar{f}^{\text{fin}}_A(x_A) = [\![f_A(x_A) > -\infty]\!]$ is J-consistent. Optionally, any time a constant can be added to a constraint and subtracted from another constraint.

Next we give important properties of the algorithm.

Theorem 12. *In any fixed point φ of Algorithm 3, $\lceil f^\varphi \rceil$ is J-consistent.*

Proof: Show that $\max_{x_{A\setminus B}} f_A(x_A) = f_B(x_B)$ implies $\bigvee_{x_{A\setminus B}} \lceil f_A \rceil(x_A) = \lceil f_B \rceil(x_B)$, which is easy. □

6. In particular, in any fixed point of the TRW-S algorithm [19], the states and state pairs whose max-marginals are maximal in trees form an arc consistent CSP. This is called *weak tree agreement* in [19].

Algorithm 3 (max-sum diffusion, parameterized)

loop
  for $(A,B) \in J$, $x_B \in X_B$ s.t. $f_B(x_B) > -\infty$ **do**
    $\varphi_{A,B}(x_B) \leftarrow \varphi_{A,B}(x_B) + [f^\varphi_B(x_B) - \max_{x_{A\setminus B}} f^\varphi_A(x_A)]/2$
  end for
end loop

Theorem 13. *If the J-consistency closure of $\lceil f^\varphi \rceil$ is initially empty then after a finite number of iterations of Algorithm 3, the upper bound strictly decreases.*

Theorem 14. *If the J-consistency closure of $\lceil f^\varphi \rceil$ is initially non-empty then:*

- *after any number of iterations of Algorithm 3, the upper bound does not change;*
- *after a finite number of iterations of Algorithm 3, $\lceil f^\varphi \rceil$ becomes the J-consistency closure of the initial $\lceil f^\varphi \rceil$.*

Proof: Theorems 13 and 14 can be proved by noting that what diffusion does to the active joint states is precisely what Algorithm 2 does to the permitted joint states. See [8, Theorem 7], cf. [19, Theorem 3.4]. □

Max-sum diffusion is not yet fully understood, in particular its convergence theory is missing. For binary WCSPs, it has been conjectured [4], [5] that diffusion converges to a fixed point, when (20) holds for all $(A,B) \in J$. Though firmly believed true, this conjecture has been never proved. We state it as follows.

Conjecture 15. *In Algorithm 3, the sequence of numbers $f^\varphi_B(x_B) - \max_{x_{A\setminus B}} f^\varphi_A(x_A)$ converges to zero.*

6.3 Handling High-arity and Global Constraints

A constraint f_A can be represented either by explicitly storing the values $f_A(x_A)$ for all $x_A \in X_A$ or by a black-box function. In constraint programming, this is known as *extensional* and *intensional* representation, respectively. For high-arity constraints, only intensional representation is possible because the set X_A is intractably large. Intensionally represented constraints of a non-fixed arity (not necessarily depending on all the variables) are referred to as *global constraints* [47], [36], [48].

The propagator of a local consistency that is trivial to execute for a low-arity constraint may be intractable for a high-arity intensional constraint. A lot of research has been done to find polynomial-time propagators for global constraints [47]. Usually, the strongest local consistency for which such a propagator is found is GAC.

Analogically, max-sum diffusion can handle an intensionally represented constraint f_A of an arbitrarily high arity if a polynomial algorithm exists to compute $\max_{x_{A\setminus B}} f^\varphi_A(x_A)$. Recall from §4 that the iteration of Algorithm 1 or 2 is the propagator for J-consistency. In this sense, the max-sum diffusion iteration can be called a *soft propagator* (for the augmenting DAG / VAC algorithm, we would need a slightly different soft propagator). Thus, soft high-arity and global constraints can be handled in the way similar to how crisp global constraints are commonly handled in CSP.

Example 5. Let $E = \binom{V}{1} \cup E' \cup (V)$ where $E' \subseteq \binom{V}{2}$. Let $J = I_{\mathrm{GAC}}(E)$. Algorithm 3 does two kinds of updates: between binary and unary constraints, and between the global and unary constraints. For the latter, we need to compute $\max_{x_{V\setminus(u)}} f^\varphi_V(x_V)$ for every $u \in V$ and $x_u \in X_u$. From (9) we have

$$f^\varphi_V(x_V) = f_V(x_V) + \sum_{v\in V} \varphi_{V,v}(x_v) \tag{23}$$

Note that (23) is an objective function of a WCSP with the global and unary constraints. Depending on f_V, computing $\max_{x_{V\setminus(u)}} f^\varphi_V(x_V)$ may or may not be tractable.

As a tractable example, let $X_V = \{0,1\}^V$ and

$$f_V(x_V) = \begin{cases} 0 & \text{if } \sum_{v\in V} x_v = n \\ -\infty & \text{otherwise} \end{cases} \tag{24}$$

be the *cardinality constraint*[7], which enforces the number of variables with state 1 to be n. Instead of the max-marginal $\max_{x_{V\setminus(u)}} f^\varphi_V(x_V)$, for simplicity we will only show how to compute $\max_{x_V} f^\varphi_V(x_V)$. It can be rewritten as a constrained maximization,

$$\max_{x_V} f^\varphi_V(x_V) = \max\Big\{ \sum_{v\in V} \varphi_{V,v}(x_v) \;\Big|\; x_V \in X_V,\, \sum_{v\in V} x_v = n \Big\}$$

One verifies that this equals $\beta + \sum_{v\in V} \varphi_{V,v}(0)$ where β is the sum of n greatest numbers from $\{\, \varphi_{V,v}(1) - \varphi_{V,v}(0) \mid v \in V \,\}$ [49]. This can be done efficiently using a dynamically updated sorted list.

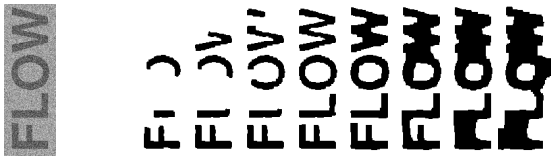
As an evidence that the approach yields plausible approximations, we present a toy experiment with image segmentation. The first image in Figure 1 is the input binary image corrupted with additive Gaussian noise. We set $f_{uv}(x_u, x_v) = [\![x_u = x_v]\!]$ and $f_v(x_v) = -[\theta(x_v) - g_v]^2$ where $\theta(x)$ is the expected intensity of a pixel with label x and g_v is the actual intensity of pixel v. We ran diffusion until the greatest residual was 10^{-8} and then we obtained x_V by taking the active state in each variable (this means, the constraint $\sum_v x_v = n$ may be satisfied only approximately). The binary images in Figure 1 show the results for different n. □

Note that e.g. all algorithms in [49] can be used as soft GAC-propagators. We anticipate that in the future, more global constraints with tractable soft propagators and useful in applications will be discovered.

7 Supermodular Problems

Supermodular constraints form the only known interesting tractable class of weighted constraints languages [11]. For binary supermodular WCSPs, it is known that

7. Binary supermodular WCSPs with cardinality constraint (24) (and its soft versions) can be well approximated by a more efficient algorithm using parametric max-flow. In detail, we observed experimentally (but did not prove) that constraining the variables $\varphi_{V,v}(x_v)$ to be equal for all $v \in V$ does not change the least upper bound. However, this of course may not hold for other global constraints.



$n =$	2000	3000	4000	5000	6000	7000	8000	9000
$\sum_v x_v =$	2008	3004	4011	5006	6004	7024	7982	9032

Fig. 1. Image segmentation with cardinality constraint.

the LP relaxation [3] is tight [50], [8]. This has been generalized to n-ary supermodular WCSPs by Werner [9] and Cooper et al. [12]. Moreover, [9], [12] show that to solve a supermodular WCSP it suffices to find any *local* optimum of the bound such that $\lceil f^\varphi \rceil$ is (G)AC.

D. Schlesinger [15] showed that binary supermodular WCSPs can be solved in polynomial time even after an unknown permutation of states in each variable. As pointed out in [15], [12], this can be done also for n-ary supermodular WCSPs.

Revisiting [50], [8], [9], [12], [15], we show in this section how to solve permuted n-ary supermodular WCSPs.

Let each domain X_v be endowed with a total order $\leq_v$. A function $f_A\colon X_A \to \bar{\mathbb{R}}$ is **supermodular** iff

$$f_A(x_A \wedge y_A) + f_A(x_A \vee y_A) \geq f_A(x_A) + f_A(y_A)$$

for any $x_A, y_A \in X_A$, where $\wedge$ ($\vee$) denotes the component-wise minimum (maximum) w.r.t. orders $\leq_v$.

Suppose that max-sum diffusion (or the augmenting DAG / VAC algorithm, §6.1) with $J = I_{\mathrm{GAC}}(E)$ found φ such that $\lceil f^\varphi \rceil$ is GAC. It can be verified that supermodularity of f_A is preserved by reparameterizations (8) on pairs[8] $(A, (v))$, hence constraints f_A^φ are supermodular too. It remains to prove the following theorem.

Theorem 16. *Let a WCSP be such that its constraints are supermodular and the CSP formed by its active joint states is GAC. Then (10) holds with equality and a maximizer of (1) can be found in polynomial time, without taking into account the orders $\leq_v$ of variable states.*

Proof: A relation $\bar{f}_A\colon X_A \to \{0, 1\}$ is a **lattice** iff

$$\bar{f}_A(x_A \wedge y_A) \wedge \bar{f}_A(x_A \vee y_A) \geq \bar{f}_A(x_A) \wedge \bar{f}_A(y_A)$$

for any $x_A, y_A \in X_A$, i.e., iff $\bar{f}_A(x_A) = \bar{f}_A(y_A) = 1$ implies $\bar{f}_A(x_A \wedge y_A) = \bar{f}_A(x_A \vee y_A) = 1$. A CSP in which each relation is a lattice is a **lattice CSP**. The lattice CSP is both max-closed and min-closed [34, §8.4.2], hence tractable. Its instance is satisfiable iff its GAC closure is not empty.

The maximizers of a supermodular function on a distributive lattice form a sublattice of this lattice [51]. Hence, $\lceil f^\varphi \rceil$ is a lattice CSP. Because $\lceil f^\varphi \rceil$ is GAC and non-empty, it is satisfiable and its solutions are in one-to-one correspondence with the maximizers of (1).

We will show how to find a solution to a lattice CSP that is GAC and non-empty. If the orders $\leq_v$ are known, a solution x_V is formed simply by the lowest (w.r.t. $\leq_v$) permitted state x_v in each variable [34], [50], [8].

If the orders $\leq_v$ are unknown, we give an algorithm to find a solution independently on them. It is easy to prove that the GAC closure of a lattice CSP is again a lattice CSP. Let us pick any v and x_v and set $\bar{f}_v(x_v) \leftarrow 0$. This can be seen as adding a unary relation to the CSP. Since any unary relation is trivially a lattice, if we now enforce GAC we again obtain a lattice CSP. This CSP is either empty or non-empty. If it is empty (i.e., x_v was the last satisfiable state in variable v), we undo the enforcing of GAC and pick a different v and x_v. If it is non-empty, we have a lattice CSP with fewer permitted states that is non-empty and GAC, hence satisfiable. We can pick another v and x_v and repeat the iteration, until each variable has a single permitted state. □

8 Incrementally Tightening Relaxation

We have seen in §3.3 that choosing different coupling schemes $J \subseteq I(E)$ yields a hierarchy of LP relaxations. Here we show that the relaxation can be tightened *incrementally*, by progressively enlarging J.

Any time during max-sum diffusion, we can extend the current J by any $J' \subseteq I(E)$ (i.e., we set $J \leftarrow J \cup J'$). This means, we add dual variables $\varphi_{A,B}$ for $(A, B) \in J'$ and set them to zero. Clearly, this does not change the current upper bound. By Theorem 11, the future diffusion iterations either preserve or improve the bound. If the bound does not improve, all we have lost is the memory occupied by the added dual variables.

Alternatively, this can be imagined as if the dual variables $\varphi_{A,B}$ were initially present for all $(A, B) \in I(E)$ but were 'locked' to zero except for those given by J. Extending J 'unlocks' some dual variables.

This scheme can be run also with other bound-optimizing algorithms (§6.1). If the algorithm is monotonic, the resulting incremental scheme is monotonic too.

The incremental scheme can be seen as a *cutting plane algorithm* because an extension of J that leads to a better bound corresponds to adding linear inequalities that separate the solution μ optimal in the current feasible polytope $P(E, X_V, J)$ from the polytope $P(E, X_V, I(E))$. Finding cutting plane(s) that separate current μ from $P(E, X_V, I(E))$ is known as the *separation problem*[9].

Note that our algorithm runs in the dual rather than primal space and that many rather than one cutting plane are added at a time: extending J by already a single $(A, B) \in I(E)$ may result in several planes intersecting $P(E, X_V, J)$, induced by the primal constraints.

8. I thank Martin Cooper for pointing out that transformation (8) preserves supermodularity only if $|B| = 1$. It is not clear whether diffusion solves supermodular WCSPs if $J \supset I_{\mathrm{GAC}}(E)$.

9. The term *separation problem* is not fully justified here because it will be applied also to the case when extending J gets the bound out of a *local* optimum (see Theorem 10), as shown later in Example 11.

8.1 Separation Test

Let us ask whether adding a given $J' \subseteq I(E)$ to current J would lead a bound improvement. We refer to this as the *separation test*. Of course, this test must be simpler and provide more insight than actually adding J' and running the bound-optimizing algorithm.

One easily invents a *sufficient* separation test: if running diffusion such that only pairs $(A, B) \in J'$ are visited improves the bound (where the amount of improvement is a good heuristic to assess the usefulness of J' [29]) then running diffusion on pairs $(A, B) \in J \cup J'$ would obviously improve the bound too. Unfortunately, this test is not *necessary*, by Example 11 given later in §9.5.

Theorems 13+14 yield a sufficient and necessary test:

Proposition 17. *Extending J by J' leads to a bound improvement iff the $(J \cup J')$-consistency closure of $\lceil f^\varphi \rceil$ is empty.*

By Proposition 17, to find out whether extending J and running Algorithm 3 improves the bound, we can extend J and run (simpler and faster) Algorithm 2.

9 Adding Zero Constraints

So far, we have considered only equivalent transformations that preserve the hypergraph (i.e., $E = E'$ in Definition 1). Let us turn to equivalent transformations that change the hypergraph. The simplest such transformation is obtained by *adding a zero constraint*, i.e., by adding a hyperedge $A \notin E$ to E, setting $f_A = 0$ (where 0 denotes the zero function), and extending J to couple A to (some or all of) the existing incident hyperedges. More complex such transformations are obtained as the composition of reparameterizations and adding zero constraints. Since adding zero constraints enables previously impossible reparameterizations, it may improve the relaxation.

All the results obtained in §3–§8 of course apply also to zero constraints. Further in §9 we discuss some specific properties of WCSPs containing zero constraints.

9.1 Complete Hierarchy of LP Relaxations

Given a WCSP with hypergraph E, its hypergraph can be completed to the complete hypergraph 2^V by adding zero constraints with scopes $2^V \setminus E$. Now, the relaxation is determined by the coupling scheme $J \subseteq I(2^V)$ alone. The zero constraints not present in any pair $(A, B) \in J$ are only virtual, they have no effect and can be ignored.

As in §3.3, relaxations for various $J \subseteq I(2^V)$ form a hierarchy, partially ordered by inclusion on $I(2^V)$. For the lowest element of the hierarchy, $J = \emptyset$, formula (9) permits no reparameterizations at all and the optimum of the LP is simply the upper bound (10). The highest element of the hierarchy, $J = I(2^V)$, yields the exact solution; however, by Proposition 5 the exact solution is obtained already for $J = I_{\mathrm{SAT}}(E)$. In between, there is a range of intermediate relaxations, including $J = I(E)$.

9.2 Adding Zero Constraints = Lifting + Projection

Let zero constraints with scopes $F \subseteq 2^V \setminus E$ be added to a WCSP (V, E, X_V, f) and let $J \subseteq I(E \cup F)$. Since zero constraints do not affect the objective function of the primal LP, the primal LP can be written as

$$\max \left\{ f^\top \mu \mid \mu \in \pi_{T(E,X_V)} P(E \cup F, X_V, J) \right\} \tag{25}$$

where $\pi_{D'} Y \subseteq \mathbb{R}^{D'}$ denotes the **projection** of a set $Y \subseteq \mathbb{R}^D$ onto dimensions $D' \subseteq D$ (i.e., $\pi_{D'}$ deletes components $D \setminus D'$ of every element of Y). Thus, zero constraints manifest themselves as a projection of the primal feasible polytope onto the space of non-zero constraints. In turn, adding zero constraints with scopes F then means *lifting* the primal feasible set from dimensions $T(E, X_V)$ to dimensions $T(E \cup F, X_V)$, imposing new primal constraints (2b)+(2c)+(2d) in the lifted space, and projecting back onto dimensions $T(E, X_V)$.

Suppose zero constraints with scopes $2^V \setminus E$ have been added. Similarly to (12), for any $J_1, J_2 \subseteq I(2^V)$ we have[10]

$$J_1 \supseteq J_2 \implies \pi_{T(E,X_V)} P(2^V, X_V, J_1) \subseteq \pi_{T(E,X_V)} P(2^V, X_V, J_2)$$

In [16], [17], [18], Wainwright et al. introduced the *marginal polytope*, formed by collections (associated with E and X_V) of marginals of some global distribution μ_V. Of fundamental importance is the marginal polytope of the complete hypergraph 2^V, given by $P(2^V, X_V, I(2^V))$. The marginal polytope of a hypergraph $E \subseteq 2^V$ is then $\pi_{T(E,X_V)} P(2^V, X_V, I(2^V))$. Therefore, for any $J \subseteq I(2^V)$, polytope $\pi_{T(E,X_V)} P(2^V, X_V, J)$ is a polyhedral *outer bound* of the marginal polytope associated with (E, X_V).

It is not hard to show [9] that the marginal polytope is the *WCSP integral hull*, i.e., the convex hull of (integral) points feasible to the integer LP given by Theorem 1.

9.3 Handling Zero Constraints in Max-sum Diffusion

In the sense of §6.3, a zero constraint can be understood as a trivial intensionally represented constraint and the max-sum diffusion iteration as its soft propagator. Let us see how zero constraints can be handled in diffusion.

Suppose $f_A = 0$. Then reparameterization (9) reads

$$f^\varphi_A(x_A) = -\sum_{B \mid (B,A) \in J} \varphi_{B,A}(x_A) + \sum_{B \mid (A,B) \in J} \varphi_{A,B}(x_B) \tag{26}$$

Example 6. Let $E = \{(2), (1,2), (2,3)\}$, $f_2 = 0$, $J = I(E)$. Then $f^\varphi_2(x_2) = -\varphi_{12,2}(x_2) - \varphi_{23,2}(x_2)$. □

More interesting is the case when there is no B such that $(B, A) \in J$. Then the first sum in (26) is vacuous and f^φ_A is the objective function of a WCSP with variables A, hypergraph $E_A = \{ B \mid (A, B) \in J \}$, and constraints $\varphi_{A,B}$. Computing $\max_{x_{A \setminus B}} f^\varphi_A(x_A)$ means *solving a WCSP*

10. Recall (Footnote 3) that $J_1 \neq J_2$ may yield the same relaxation. If all the constraints are non-zero, this happens iff $P(2^V, X_V, J_1) = P(2^V, X_V, J_2)$. If constraints with scopes $2^V \setminus E$ are zero, this happens iff $\pi_{T(E,X_V)} P(2^V, X_V, J_1) = \pi_{T(E,X_V)} P(2^V, X_V, J_2)$. Note that the former condition implies the latter one but not *vice versa*.

on a smaller hypergraph[11]. Adding a zero constraint f_A makes sense only if WCSPs on E_A are easier to solve than the WCSP on E. Note that no function of arity $|A|$ needed to be explicitly stored.

Example 7. Let $V = (1,2,3,4)$, $E = \{\,(1),\ (2),\ (3),\ (4),\ (1,2),\ (2,3),\ (3,4),\ (1,4),\ (1,3),\ V\,\}$, $f_V = 0$, and $J = \{\,(V,(1,2)),\ (V,(2,3)),\ (V,(3,4)),\ (V,(1,4))\,\}$. Then E_A is a cycle of length 4 and $f_V^{\varphi}(x_1,x_2,x_3,x_4) = \varphi_{V,12}(x_1,x_2) + \varphi_{V,23}(x_2,x_3) + \varphi_{V,34}(x_3,x_4) + \varphi_{V,14}(x_1,x_4)$. □

Example 8. In this example, we show how adding short cycles improves relaxations of binary WCSPs.

We tested two types of graphs:

- $E = \binom{V}{1} \cup E'$ where $E' \subseteq \binom{V}{2}$ is the 2-dimensional 4-connected $m \times m$ grid.
- Complete graph, $E = \binom{V}{1} \cup \binom{V}{2}$ where $|V| = m$.

For the grid graph, we tested two relaxations: $J_1 = I(E)$ and $J_2 = I(E \cup F)$ where $F \subseteq \binom{V}{4}$ contains all hyperedges A such that $E \cap 2^A$ is a cycle of length 4 (as in Example 7). For the complete graph, we tested two relaxations: $J_1 = I(E)$ and $J_2 = I\big(E \cup \binom{V}{3}\big)$, i.e., the relaxation J_2 was obtained by adding all 3-cycles.

Each variable had the same number of states, $|X_v|$. We tested five types of constraints f:

- `random`: all weights $f_v(x_v)$ and $f_{uv}(x_u,x_v)$ were i.i.d. drawn from the normal distribution $\mathcal{N}[0;1]$.
- `attract`: $f_{uv}(x_u,x_v) = [\![x_u = x_v]\!]$ and $f_v(x_v)$ were drawn from $\mathcal{N}[0;1.6]$. We chose variance 1.6 because it yielded (by trial) the hardest instances.
- `repulse`: $f_{uv}(x_u,x_v) = [\![x_u \neq x_v]\!]$ and $f_v(x_v)$ were drawn from $\mathcal{N}[0;0.1]$.

In the other types (apply only to grid graphs), $f_v(x_v)$ were drawn from $\mathcal{N}[0;1]$ and the binary constraints were crisp, $f_{uv}(x_u,x_v) \in \{-\infty, 0\}$. They were taken from [7]:

- `lines`: $f_{uv}(x_u,x_v)$ were as in [7, Figure 19a].
- `curve`: $f_{uv}(x_u,x_v)$ were as in [7, Figure 15a].

On a number of WCSP instances, we counted how many instances were solved to optimality. Once diffusion converged, the instance was marked as solved if there was a unique active state in each variable. Table 1 shows the results, where r_1 resp. r_2 is the proportion of instances solved to optimality by relaxation J_1 resp. J_2. There were 100 trials for each line; in each trial, we randomly drew instances from the instance type and computed relaxation J_1 until it was not tight, and then we computed relaxation J_2. Runtime for `random` or `attract` on 100×100 grid and $|X_v| = 4$ was several minutes (for a non-optimized Matlab+C code).

For `random` and `attract` on both graphs and for `repulse` on grids, relaxation J_2 was much tighter and was often exact even for large graphs. For crisp binary constraints (on grids), relaxation J_2 clearly beat J_1, but for $m \geq 25$ `lines` and `curve` were unsolvable. This is not too surprising because `lines` and `curve` are much harder than instances typical in low-level vision (such as the benchmarks in [52]). In more detail [7], they are easy if the data terms $f_v(x_v)$ are 'close to a feasible image' but this is not at all the case if $f_v(x_v)$ are random.

graph	constraints	m	$\lvert X_v\rvert$	r_1	r_2
grid	`random`	15	5	0.01	1.00
grid	`random`	25	3	0.00	0.98
grid	`random`	100	3	0.00	0.72
grid	`attract`	15	5	0.79	0.99
grid	`attract`	25	5	0.48	0.98
grid	`attract`	100	5	0.00	0.81
grid	`repulse`	10	3	0.18	1.00
grid	`repulse`	20	3	0.00	0.98
grid	`repulse`	50	3	0.00	0.57
grid	`lines`	10	4	0.71	0.85
grid	`lines`	15	4	0.40	0.54
grid	`lines`	25	4	0.00	0.05
grid	`curve`	10	9	0.17	0.65
grid	`curve`	15	9	0.00	0.24
grid	`curve`	25	9	0.00	0.00
complete	`random`	10	3	0.01	1.00
complete	`random`	15	3	0.00	0.89
complete	`random`	20	3	0.00	0.40
complete	`random`	25	2	0.00	0.87
complete	`repulse`	4	2	0.00	0.98
complete	`repulse`	5	2	0.00	0.00
complete	`repulse`	4	3	0.00	0.00

TABLE 1
Tightening the LP relaxation by adding short cycles.

Despite it is known that densely connected instances are hard [53], it is surprising that `repulse` was never solved even on very small graphs. Note, `repulse` encourages neighboring variables to have different states, thus it is close to the difficult graph coloring problem. □

9.4 Optimality under Presence of Zero Constraints

As shown in §5, optimality of the upper bound (10) depends on the CSP formed by active joint states. Since $f_A = 0$ implies $\lceil f_A \rceil = 1$ (where 1 denotes the universal relation), adding a zero constraint to the WCSP means adding a universal relation to this CSP. After reparameterization (9), a zero constraint $f_A = 0$ becomes f_A^{φ} which is no longer zero and a universal relation $\lceil f_A \rceil = 1$ becomes $\lceil f_A^{\varphi} \rceil$ which is no longer universal.

By Theorem 8, the relaxation is tight iff $\lceil f^{\varphi} \rceil$ is satisfiable. It can happen that the CSP formed only by relations $\lceil f_A^{\varphi} \rceil$ with $f_A \neq 0$ is satisfiable but the whole CSP $\lceil f^{\varphi} \rceil$ is unsatisfiable. Example 9 shows this is indeed possible. Thus, *we must not ignore zero constraints* when testing for bound optimality and recovering an optimizer.

Example 9. First, we give an unsatisfiable ternary CSP $(V,E,X_V,\bar{f})$ whose binary part is satisfiable. Let $V = (1,2,3,4)$, $X_V = \{0,1\}^V$, $E = \binom{V}{2} \cup \binom{V}{3}$. Let $\bar{f}$ be defined by $\bar{f}_{12} = \bar{f}_{13} = \bar{f}_{14} = \bar{f}_{23} = \bar{f}_{24} = \bar{f}_{34}$ and $\bar{f}_{123} = \bar{f}_{124} = \bar{f}_{134} = \bar{f}_{234}$ where relations $\bar{f}_{123}$ and $\bar{f}_{12}$ are given by (17). Thus, the CSP consists of four copies of the CSP from Example 4 glued together[12]. Its six binary relations are

11. These sub-WCSPs roughly correspond to the subproblems in the decomposition approach [17], [19], [20] (see §1.1) and to the 'slave' problems in the dual decomposition formulation [21].

12. This CSP is the ternary generalization of the well-known binary unsatisfiable CSP on three variables (the 'frustrated cycle'), given by $\bar{f}_{12} = \bar{f}_{13} = \bar{f}_{23}$ where $\bar{f}_{12}(x_1,x_2) = \bar{x}_1 x_2 \vee x_1 \bar{x}_2$.

shown below (the ternary relations are not visualized):

For $J = I(E)$, check that $\bar{f}$ is J-consistent. Any $x_{1234} \in \{(1,0,0,0),(0,1,0,0),(0,0,1,0),(0,0,0,1),(0,0,0,0)\}$ satisfies all the six binary relations but none of them satisfies all the ternary relations. Hence, $\bar{f}$ is unsatisfiable.

Second, we give a WCSP (V,E,X_V,f) with the ternary constraints being zero and we give a diffusion fixed point φ such that $\bar{f} = \lceil f^\varphi \rceil$. This may look trivial because any CSP is indeed realizable as $\lceil f^\varphi \rceil$ for some f and φ. But we must not forget about the constraint that the ternary constraints are zero. E.g., $f_{123} = 0$ and hence, by (9), $f^\varphi_{123} = \varphi_{123,12} + \varphi_{123,13} + \varphi_{123,23}$, i.e., f^φ_{123} must be a sum of binary functions. We show such f and φ exist.

Let f and φ be defined by $f_{123} = f_{124} = f_{134} = f_{234} = 0$, $f_{12} = f_{13} = f_{14} = f_{23} = f_{24} = f_{34}$ and $\varphi_{123,12} = \varphi_{123,13} = \varphi_{123,23} = \varphi_{124,12} = \varphi_{124,14} = \varphi_{124,24} = \varphi_{134,13} = \varphi_{134,14} = \varphi_{134,34} = \varphi_{234,23} = \varphi_{234,24} = \varphi_{234,34}$, where

$$f_{12}(x_1,x_2) = 4\bar{x}_1\bar{x}_2 + 9(\bar{x}_1x_2 + x_1\bar{x}_2) + 7x_1x_2$$
$$\varphi_{123,12}(x_1,x_2) = \bar{x}_1\bar{x}_2 + 2(\bar{x}_1x_2 + x_1\bar{x}_2)$$

From (9) we get that f^φ is given by $f^\varphi_{123} = f^\varphi_{124} = f^\varphi_{134} = f^\varphi_{234} = \varphi_{123,12} + \varphi_{123,13} + \varphi_{123,23}$ and $f^\varphi_{12} = f^\varphi_{13} = f^\varphi_{14} = f^\varphi_{23} = f^\varphi_{24} = f^\varphi_{34} = f_{12} - \varphi_{123,12} - \varphi_{124,12}$ where

$$f^\varphi_{123}(x_1,x_2,x_3) = 3\bar{x}_1\bar{x}_2\bar{x}_3 + 5(x_1\bar{x}_2\bar{x}_3 + \bar{x}_1x_2\bar{x}_3 + \bar{x}_1\bar{x}_2x_3) + 4(x_1x_2\bar{x}_3 + x_1\bar{x}_2x_3 + \bar{x}_1x_2x_3)$$
$$f^\varphi_{12}(x_1,x_2) = 5(\bar{x}_1\bar{x}_2 + \bar{x}_1x_2 + x_1\bar{x}_2) + 4x_1x_2$$

Check that $f^\varphi_B(x_B) = \max_{x_{A\setminus B}} f^\varphi_A(x_A)$ for each $(A,B) \in J$, i.e., φ is a diffusion fixed point. Check that $\bar{f} = \lceil f^\varphi \rceil$.

Note a surprising property of f^φ_{123}: function f^φ_{123} is a sum of binary functions but (see Example 4) relation $\lceil f^\varphi_{123} \rceil$ is not a conjunction of any binary relations[13]. □

9.5 Adding Zero Constraints Incrementally

We have shown in §8 how the relaxation can be tightened incrementally by extending J. When combined with adding zero constraints[14], this can be seen as a cutting plane algorithm which adds sets of linear inequalities separating μ optimal in $\pi_{T(E,X_V)}P(E,X_V,J)$ from the marginal polytope $\pi_{T(E,X_V)}P(2^V,X_V,I(2^V))$. Here we focus mainly on the separation problem.

The separation test (§8.1) asks whether extending J by $J' \subseteq I(2^V)$ would lead to bound improvement. In general, this is answered by Proposition 17. However, if J' has a special form, this can be formulated in terms of *satisfiability of a sub-CSP* of $\lceil f^\varphi \rceil$. For a CSP with hypergraph E, we define its **restriction** to $F \subseteq E$ to be the CSP with hypergraph F and the relations inherited from the original CSP.

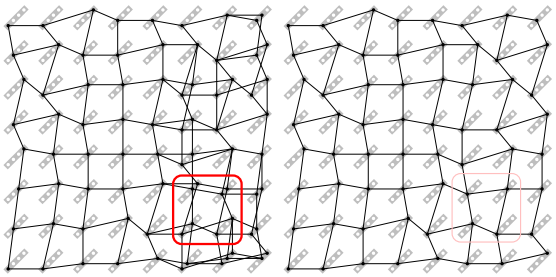


Fig. 2. Incrementally adding zero constraints.

Proposition 18. *Let (V,E,X_V,f) be a WCSP. Let $F \subseteq E$. Let us ask whether adding the zero constraint with scope $\bigcup F$, extending J by $J' = I_{\mathrm{SAT}}(F)$, and running max-sum diffusion will improve the bound.*

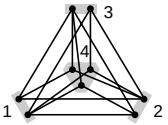
- *If the restriction of $\lceil f^\varphi \rceil$ to F is not satisfiable then the answer is 'yes'.*
- *If, in the restriction of $\lceil f^\varphi \rceil$ to F, every permitted joint state is satisfiable then the answer is 'no'.*

Proof: In Proposition 17, apply Propositions 5 and 6 on the restriction of $\lceil f^\varphi \rceil$ on hypergraph F. □

Example 10. Let us return to Example 8. Figure 2 (left) shows the CSP $\lceil f^\varphi \rceil$ after convergence of max-sum diffusion for relaxation J_1 of an instance `random` with size $m = 8$ and $|X_v| = 4$. The upper bound is not optimal because of the depicted unsatisfiable sub-CSP. Let A denote the four depicted variables. After adding the zero constraint with scope A, diffusion yielded Figure 2 (right) with an improved bound – here, the exact solution. Of course, many such steps are typically needed.

The inconsistent sub-CSP is supported only by 2 (rather than 4) states of each variable $v \in A$. Thus, instead of adding a zero constraint with 4 states, we could add a constraint with 2 states. For variables with large domains, this could drastically reduce the computational effort (see experiments in [29]). This would mean using the fine-grained hierarchy of coupling schemes[15] (7). □

Example 11. Let $\lceil f^\varphi \rceil$ be this unsatisfiable CSP:



13. This suggests an interesting problem: Given $\bar{f}_V\colon X_V \to \{0,1\}$ and $E \subseteq 2^V$, find $f\colon T(E,X_V) \to \bar{\mathbb{R}}$ such that $\bar{f}_V = \lceil \sum_{A\in E} f_A \rceil$ or show that no such f exists. For given (V,E,X_V), characterize the class of relations $\bar{f}_V$ realizable in this way.

14. The incremental scheme from §8 is not restricted to zero constraints, it can be used also with non-zero constraints. E.g., given a WCSP with hypergraph $E \cup F$ where constraints E are 'easy' (unary, binary) and constraints F are 'hard' (high arity), we can first solve constraints E and then incrementally extend J to include constraints F. In both cases, Proposition 17 applies.

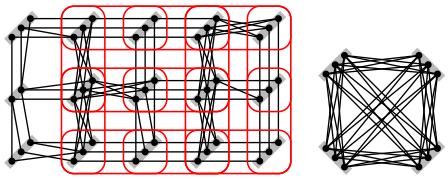
15. Using the fine-grained hierarchy of coupling schemes (i.e., using (7) rather than (2b)) would require adapting Algorithm 3 and the theorems in §6.2 because e.g. Theorem 11 does not hold. This would require some more research, for which paper [54] might be relevant.

The sub-CSP on variables $(1,2,4)$ is satisfiable but has unsatisfiable permitted joint states. Let us add constraint $f_{124} = 0$. This makes the PWC closure of the whole CSP empty. Thus, running diffusion on the sub-WCSP on variables $(1,2,4)$ will not improve the bound but running diffusion on the whole WCSP will.

As shown in [8, Figure 5b], the CSP $\lceil f^\varphi \rceil$ in the figure corresponds to a local optimum of φ that is not global. Thus, adding zero constraints sometimes can get the bound out of a local optimum. □

For some WCSP instances, it can be useful to add more complex subproblems than cycles.

Example 12. Consider the binary CSP $\lceil f^\varphi \rceil$ in figure (a), whose hypergraph E has 15 variables (i.e., 1-element hyperedges) and 22 2-element hyperedges:



(a) (b)

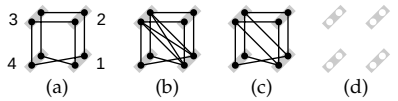
Let $J = I(E)$. The CSP is J-consistent, thus diffusion cannot improve the bound. It contains no unsatisfiable cycles but it is unsatisfiable because of the sub-CSP with hypergraph $F \subset E$ marked in red: F has $|\bigcup F| = 12$ variables and 14 2-element hyperedges. The sub-CSP is unsatisfiable because it can be reduced (by 'contracting' the identity relations) to the CSP in figure (b), which encodes the (impossible) task of 3-coloring the graph K_4.

Adding the zero constraint with scope $\bigcup F$ and extending J by $J' = I_{\mathrm{SAT}}(F) = \{(\bigcup F, A) \mid A \in F\}$ makes the J-consistency closure of $\lceil f^\varphi \rceil$ empty (verify by Algorithm 2). Hence, diffusion will now improve the bound. Computing $\max_{x_{A\setminus B}} f^\varphi_A(x_A)$ for $A = \bigcup F$ needs more effort than if F was a cycle, but is still feasible. □

Given a family $\mathcal{J}$ of tentative subsets of $I(2^V)$, the separation problem consists in finding a 'small' $J' \in \mathcal{J}$ that will improve the bound. If $\mathcal{J}$ is small, J' can be searched exhaustively. If $\mathcal{J}$ has an intractable size, Proposition 18 translates the separation problem to finding an unsatisfiable sub-CSP of $\lceil f^\varphi \rceil$. It is subject to future research to discover polynomial-time algorithms to find an unsatisfiable sub-CSP of a CSP, where the sub-CSP are from some combinatorially large class (such as cycles). Finding *minimal* unsatisfiable sub-CSPs (though not in polynomial time) has been addressed in [55], [54].

Finally, note that extending J by elements of $\mathcal{J}$ one by one has a theoretical problem: it can happen that adding any single element of $\mathcal{J}$ does not improve the bound but adding the union of several elements of $\mathcal{J}$ does.

Example 13. Consider the CSP with $E = \{(1), (2), (3), (4), (1,2), (2,3), (3,4), (1,4)\}$ in figure (a):



(a) (b) (c) (d)

Adding simultaneously zero constraints with scopes $(1,3)$, $(1,2,3)$, $(1,3,4)$ makes the PWC closure empty (figures b,c,d). However, adding any of these constraints separately does not make the PWC closure empty. □

9.6 k-consistency of Active Joint States

If E is closed to intersection and $J = I(E)$, $\lceil f^\varphi \rceil$ can always be made PWC by changing φ. PWC is the strongest local consistency of $\lceil f^\varphi \rceil$ achievable without adding zero constraints. By adding zero constraints, stronger local consistencies of $\lceil f^\varphi \rceil$ can be achieved.

By §4.2, adding all possible zero constraints of arity k and $k-1$ and running max-sum diffusion with $J = I(E)$ makes $\lceil f^\varphi \rceil$ k-consistent. Unlike in CSP, the previously added k-ary constraints *cannot* be removed after this [39, §8]. Thus, there is a difference in the rôle of k-consistency in CSP and WCSP. Enforcing strong k-consistency of a CSP requires adding only relations of arity less than k. For a WCSP, enforcing strong k-consistency of $\lceil f^\varphi \rceil$ requires adding constraints of arity less *or equal* to k. E.g., a binary CSP can be made 3-consistent and remain binary; for a binary WCSP, $\lceil f^\varphi \rceil$ can be made 3-consistent but only at the expense of making the WCSP ternary[16].

Similarly as in CSP (§4.2), strong k-consistency of $\lceil f^\varphi \rceil$ can be enforced in a more efficient way, by incrementally adding only *some* of all missing constraints of arity k or less. Supposing $\lceil f^\varphi \rceil$ is $(k-1)$-consistent, it is made k-consistent as follows. Whenever $|A| = k-1$ and x_A cannot be extended to some variable v, we add constraint $f_{A\cup(v)} = 0$, and if $A \notin E$ we also add constraint $f_A = 0$. Then we set $J = I(E)$ and re-optimize the bound.

As making $\lceil f^\varphi \rceil$ 3-consistent requires adding new ($O(|V|^3)$ at worst) binary and ternary constraints, it is practical only for mid-size instances. Otherwise, partial forms of 3-consistency can be considered. One such form is suggested by Theorem 4: add only edges to E that make E chordal[17] (rather than complete). Since this can be still prohibitive, even fewer edges can be added.

Example 14. Let E be the $m \times m$ grid graph. We did a 'partial chordal completion' of E as follows: of all edges necessary to complete E to a chordal graph, we added only those edges (u,v) for which the Manhattan distance between nodes u and v in the original graph was not greater than d (this can be done by a simple modification of known algorithms for chordal completion). Then we triangulated the graph, added the resulting triangles and

16. Note, otherwise we'd get a paradox. A binary CSP with Boolean variables is tractable: it is satisfiable iff enforcing 3-consistency does not make it empty [34, §8.4.2]. If the active joint states of any binary WCSP with Boolean variables could be made 3-consistent without adding ternary constraints, we would have a polynomial algorithm to solve any binary WCSP with Boolean variables, which is intractable.

17. It is well-known that chordal completion is done also before the junction tree algorithm [32]. This is unrelated to its purpose here.

ran max-sum diffusion. The table shows the proportions of instances drawn from type `lines` (see Example 8) that were solved to optimality for various m and d. The number of added triangles is stated in parentheses.

	$d=1$	$d=2$	$d=3$	$d=4$	$d=5$
m=10	0.71 (0)	0.85 (162)	0.88 (313)	0.92 (592)	0.95 (703)
m=15	0.40 (0)	0.54 (392)	0.58 (785)	0.77 (1469)	0.88 (1906)
m=20	0.11 (0)	0.26 (722)	0.28 (1483)	0.38 (2773)	0.57 (3167)
m=25	0.00 (0)	0.05 (1152)	0.06 (2392)	0.16 (4459)	0.33 (5819)

Comparing to Table 1 shows that the added triangles significantly tightened the relaxation. We remark that using the above described more efficient incremental algorithm would results in fewer added triangles. □

9.6.1 3-consistency and the Cycle Inequalities

By Theorem 4, in a 3-consistent binary CSP every cycle is consistent[18]. This closely resembles the algorithm by Barahona and Mahjoub [31] (applied to WCSP in [26]) to separate cycle inequalities in the cut polytope [30]. While the algorithm [31] is primal and works only for Boolean variables, enforcing 3-consistency of $\lceil f^\varphi \rceil$ works in the dual space and for variables with any number of states. The precise relationship between the algorithm [31] and enforcing 3-consistency of $\lceil f^\varphi \rceil$ has yet to be clarified.

In particular, it is known that the *planar max-cut problem* is tractable, solved by a linear program over the *semimetric polytope* defined by the cycle inequalities [30, §27.3], cf. [56], [57]. The planar max-cut problem is equivalent to the WCSP (V, X_V, E, f) where $X_V = \{0,1\}^V$, $E \subseteq \binom{V}{2}$ is a planar graph, and constraints f have the form $f_{uv}(x_u, x_v) = c_{uv}[\![x_u = x_v]\!]$ with $c_{uv} \in \mathbb{R}$ (i.e., c_{uv} have arbitrary signs). It is an open problem whether this WCSP is solved by enforcing 3-consistency of $\lceil f^\varphi \rceil$.

10 Conclusion

We have tried to pave the way to WCSP solvers that would natively handle non-binary constraints (possibly of high-arity and represented by a black-box function) and use the cutting plane strategy.

Though we have considered only max-sum diffusion, most of the theory applies to the other bound-optimizing algorithms from §6.1, most notably to the augmenting DAG / VAC algorithm. Choosing which bound-optimizing algorithm to use is important and each algorithm has pros and cons:

- Max-sum diffusion is extremely simple and very flexible. Its drawback is that it is rather slow – for images, several times than the closely related and slightly more complex TRW-S.
- The augmenting DAG / VAC algorithm is complex and painful to implement efficiently [7] but has a unique advantage in its *incrementality*: if we run it to convergence and make a 'small' change to the WCSP, it typically needs a 'small' number of iterations to re-converge. This makes it suitable for cutting plane schemes (as observed in [28]), branch&cut search, and for incremental fixing of undecided variables. All the other bound-optimizing algorithms from §6.1 need a large number of iterations to re-converge, and this does not seem possible to avoid by any clever scheduling of iterations.
- In the light of the possibility to add zero constraints, the globally optimal algorithms (such as subgradient and smoothing methods) lose something of their attractivity. It is always a question whether to use these algorithms (which are slower than the local algorithms, especially when the LP relaxation is not tight) or to add zero constraints.

We considered only obtaining upper bounds on WCSP and we have not discussed rounding schemes to round undecided variables or using the LP relaxation as part of a search, such as branch&bound [12] or branch&cut.

18. Note, this shows that if all cycles of length 3 are added to a WCSP on a complete graph in Example 8 (relaxation J_2), the relaxation cannot be further improved by adding any cycles of greater lengths.

Acknowledgments

This research was supported by the European Commission grant 215078 and the Czech government grant MSM6840770038. I thank Mikhail I. Schlesinger for open discussions on his unpublished work. Martin Cooper and Thomas Schiex provided useful remarks on WCSP.

References

[1] P. Meseguer, F. Rossi, and T. Schiex, "Soft constraints," ch. 9, in [10].

[2] P. L. Hammer, P. Hansen, and B. Simeone, "Roof duality, complementation and persistency in quadratic 0-1 optimization," *Math. Programming*, vol. 28, pp. 121–155, 1984.

[3] M. I. Shlezinger, "Syntactic analysis of two-dimensional visual signals in noisy conditions," *Cybernetics and Systems Analysis*, vol. 12, no. 4, pp. 612–628, 1976, translation from Russian: Sintaksicheskiy analiz dvumernykh zritelnikh signalov v usloviyakh pomekh, Kibernetika, vol. 12, no. 4, pp. 113-130, 1976.

[4] V. A. Kovalevsky and V. K. Koval, "A diffusion algorithm for decreasing energy of max-sum labeling problem," approx. 1975, Glushkov Institute of Cybernetics, Kiev, USSR. Unpublished.

[5] "Personal communication of the author with Mikhail I. Schlesinger," 2000-2005.

[6] V. K. Koval and M. I. Schlesinger, "Dvumernoe programmirovanie v zadachakh analiza izobrazheniy (Two-dimensional programming in image analysis problems)," *USSR Academy of Science, Automatics and Telemechanics*, vol. 8, pp. 149–168, 1976, in Russian.

[7] T. Werner, "A linear programming approach to max-sum problem: A review," Center for Machine Perception, Czech Technical University, Tech. Rep. CTU–CMP–2005–25, December 2005.

[8] ——, "A linear programming approach to max-sum problem: A review," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 29, no. 7, pp. 1165–1179, July 2007.

[9] ——, "High-arity interactions, polyhedral relaxations, and cutting plane algorithm for soft constraint optimisation (MAP-MRF)," in *Computer Vision and Pattern Recognition (CVPR) Conf., Anchorage, USA*, June 2008.

[10] F. Rossi, P. van Beek, and T. Walsh, *Handbook of Constraint Programming*. Elsevier, 2006.

[11] D. Cohen, M. Cooper, P. Jeavons, and A. Krokhin, "The complexity of soft constraint satisfaction," *Artificial Intelligence*, vol. 170, pp. 983–1016, 2006.

[12] M. C. Cooper, S. de Givry, M. Sánchez, T. Schiex, and M. Zytnicki, "Virtual arc consistency for weighted CSP," in *Conf. on Artificial Intelligence (AAAI)*, July 2008, pp. 253–258.

[13] U. Montanari, "Networks of constraints: Fundamental properties and application to picture processing," *Information Science*, vol. 7, pp. 95–132, 1974.

[14] A. Rosenfeld, R. A. Hummel, and S. W. Zucker, "Scene labeling by relaxation operations," *IEEE Trans. on Systems, Man, and Cybernetics*, vol. 6, no. 6, pp. 420–433, June 1976.

[15] D. Schlesinger, "Exact solution of permuted submodular MinSum problems," in *Conf. Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR), Ezhou, China*. Springer, 2007, pp. 28–38.

[16] M. J. Wainwright and M. I. Jordan, "Graphical models, exponential families, and variational inference," UC Berkeley, Dept. of Statistics, Tech. Rep. 649, 2003.

[17] M. Wainwright, T. Jaakkola, and A. Willsky, "MAP estimation via agreement on (hyper)trees: message passing and linear programming approaches," *IEEE Trans. Information Theory*, vol. 51, no. 11, pp. 3697–3717, 2005.

[18] M. J. Wainwright and M. I. Jordan, "Graphical models, exponential families, and variational inference," *Foundations and Trends in Machine Learning*, vol. 1, no. 1-2, pp. 1–305, 2008.

[19] V. Kolmogorov, "Convergent tree-reweighted message passing for energy minimization," *IEEE Trans. Pattern Analysis and Machine Intelligence*, vol. 28, no. 10, pp. 1568–1583, 2006.

[20] J. K. Johnson, D. M. Malioutov, and A. S. Willsky, "Lagrangian relaxation for MAP estimation in graphical models," in *Allerton Conf. Communication, Control and Computing*, 2007.

[21] N. Komodakis, N. Paragios, and G. Tziritas, "MRF optimization via dual decomposition: Message-passing revisited," in *Intl. Conf. Computer Vision (ICCV)*, 2007.

[22] D. P. Bertsekas, *Nonlinear Programming*. Belmont, MA: Athena Scientific, 1999.

[23] Y. Weiss, C. Yanover, and T. Meltzer, "MAP estimation, linear programming and belief propagation with convex free energies," in *Conf. Uncertainty in Artificial Intelligence (UAI)*, 2007.

[24] C. Rother, T. Minka, A. Blake, and V. Kolmogorov, "Cosegmentation of image pairs by histogram matching – incorporating a global constraint into MRFs," in *Conf. Computer Vision and Pattern Recognition (CVPR)*, 2006.

[25] A. Koster, C. P. M. van Hoesel, and A. W. J. Kolen, "The partial constraint satisfaction problem: Facets and lifting theorems," *Operations Research Letters*, vol. 23, no. 3–5, pp. 89–97, 1998.

[26] D. Sontag and T. Jaakkola, "New outer bounds on the marginal polytope," in *Neural Information Processing Systems (NIPS)*, 2007.

[27] M. P. Kumar and P. H. S. Torr, "Efficiently solving convex relaxations for MAP estimation," in *Intl. Conf. on Machine Learning (ICML)*. ACM, 2008, pp. 680–687.

[28] N. Komodakis and N. Paragios, "Beyond loose LP-relaxations: Optimizing MRFs by repairing cycles," in *European Conf. on Computer Vision (ECCV)*, 2008.

[29] D. Sontag, T. Meltzer, A. Globerson, T. Jaakkola, and Y. Weiss, "Tightening LP relaxations for MAP using message passing," in *Conf. Uncertainty in Artificial Intelligence (UAI)*, 2008.

[30] M. M. Deza and M. Laurent, *Geometry of Cuts and Metrics*. Springer, Berlin, 1997.

[31] F. Barahona and A. R. Mahjoub, "On the cut polytope," *Mathematical Programming*, vol. 36, no. 2, pp. 157–173, 1986.

[32] S. Lauritzen, *Graphical Models*. Oxford University Press, 1996.

[33] A. Mackworth, "Constraint satisfaction," in *Encyclopaedia of Artificial Intelligence*. Wiley, 1991, pp. 285–292.

[34] D. Cohen and P. Jeavons, "The complexity of constraint languages," ch. 8, in [10].

[35] J.K.Pearson and P.G.Jeavons, "A survey of tractable constraint satisfaction problems," Royal Holloway, University of London, Tech. Rep. CSD-TR-97-15, July 1997.

[36] C. Bessiere, "Constraint propagation," ch. 3, in [10].

[37] R. Debruyne and C. Bessière, "Domain filtering consistencies," *Journal of Artificial Intelligence Research*, no. 14, pp. 205–230, 2001.

[38] E. C. Freuder, "A sufficient condition for backtrack-free search," *J. ACM*, vol. 29, no. 1, pp. 24–32, 1982.

[39] T. Werner, "Marginal consistency: Unifying constraint propagation on commutative semirings," in *Intl. Workshop on Preferences and Soft Constraints (co-located with Conf. on Principles and Practice of Constraint Programming)*, September 2008, pp. 43–57.

[40] C. Bliek and D. Sam-Haroud, "Path consistency on triangulated constraint graphs," in *Intl. Joint Conference on Artificial Intelligence (IJCAI)*, 1999, pp. 456–461.

[41] M. I. Schlesinger, "False minima of the algorithm for minimizing energy of max-sum labeling problem," 1976, Glushkov Institute of Cybernetics, Kiev, USSR. Unpublished.

[42] V. N. Kolmogorov and M. J. Wainwright, "On the optimality of tree-reweighted max-product message-passing," in *Conf. Uncertainty in Artificial Intelligence (UAI)*, 2005.

[43] A. Globerson and T. Jaakkola, "Fixing max-product: Convergent message passing algorithms for MAP LP-relaxations," in *Neural Information Processing Systems (NIPS)*, 2008, pp. 553–560.

[44] T. Werner and A. Shekhovtsov, "Unified framework for semiring-based arc consistency and relaxation labeling," in *12th Computer Vision Winter Workshop, St. Lambrecht, Austria*. Graz University of Technology, February 2007, pp. 27–34.

[45] M. I. Schlesinger and V. V. Giginjak, "Solving (max,+) problems of structural pattern recognition using equivalent transformations," *Upravlyayushchie Sistemy i Mashiny (Control Systems and Machines), Kiev, Naukova Dumka*, vol. 1 and 2, 2007, in Russian, English translation available on www.

[46] P. Ravikumar, A. Agarwal, and M. J. Wainwright, "Message-passing for graph-structured linear programs: proximal projections, convergence and rounding schemes," in *Intl. Conf. on Machine Learning (ICML)*. ACM, 2008, pp. 800–807.

[47] W.-J. van Hoeve and I. Katriel, "Global constraints," ch. 7, in [10].

[48] C. Bessière and P. V. Hentenryck, "To be or not to be ... a global constraint," in *Conf. on Principles and Practice of Constraint Programming (CP)*, 2003, pp. 789–794.

[49] R. Gupta, A. A. Diwan, and S. Sarawagi, "Efficient inference with cardinality-based clique potentials," in *Intl. Conf. on Machine Learning (ICML)*, 2007, pp. 329–336.

[50] M. I. Schlesinger and B. Flach, "Some solvable subclasses of structural recognition problems," in *Czech Pattern Recognition Workshop*. Czech Pattern Recognition Society, 2000.

[51] D. M. Topkis, *Supermodularity and Complementarity*, ser. Frontiers of Economic Research. Princeton University Press, 1998.

[52] R. Szeliski, R. Zabih, D. Scharstein, O. Veksler, V. Kolmogorov, A. Agarwal, M. Tappen, and C. Rother, "A comparative study of energy minimization methods for Markov random fields," in *Eur. Conf. Computer Vision (ECCV)*, 2006, pp. II: 16–29.

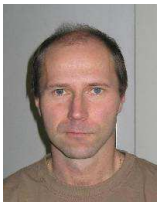
[53] V. Kolmogorov and C. Rother, "Comparison of energy minimization algorithms for highly connected graphs," in *European Conf. Computer Vision (ECCV)*, 2006, pp. II: 1–15.

[54] E. Gregoir, B. Mazure, and C. Piette, "MUST: Provide a finer-grained explanation of unsatisfiability," in *Conf. on Principles and Practice of Constraint Programming (CP)*, 2007, pp. 317–331.

[55] F. Hemery, C. Lecoutre, L. Sais, and F. Boussemart, "Extracting MUCs from constraint networks," in *Europ. Conf. on Artificial Intelligence (ECAI)*, Trento, Italy, 2006, pp. 113–117.

[56] A. Globerson and T. Jaakkola, "Approximate inference using planar graph decomposition," in *Neural Information Processing Systems (NIPS)*, 2006, pp. 473–480.

[57] N. N. Schraudolph and D. Kamenetsky, "Efficient exact inference in planar Ising models," in *Neural Information Processing Systems (NIPS)*, 2008, pp. 1417–1424.



Tomáš Werner spent most of his time at the Center for Machine Perception, a research group at the Czech Technical University in Prague. In 1999, he defended there his PhD thesis on multiple view geometry in computer vision. Since 2000, he spent 1.5 years in the Visual Geometry Group at the Oxford University, U.K., and then returned to Prague. Until 2002 his main interest was multiple view geometry and since then, optimization and Markov random fields.

1 Cutting plane methods in machine learning

Vojtěch Franc xfrancv@cmp.felk.cvut.cz
Czech Technical University in Prague
Technická 2, 166 27 Prague 6
Czech Republic

Sören Sonnenburg Soeren.Sonnenburg@tu-berlin.de
Berlin Institute of Technology
Franklinstr. 28/29
10587 Berlin, Germany

Tomáš Werner werner@cmp.felk.cvut.cz
Czech Technical University in Prague
Technická 2, 166 27 Prague 6
Czech Republic

Cutting plane methods are optimization techniques that incrementally construct an approximation of a feasible set or an objective function by linear inequalities, called cutting planes. Numerous variants of this basic idea are among standard tools used in convex nonsmooth optimization and integer linear programing. Recently, cutting plane methods have seen growing interest in the field of machine learning. In this chapter, we describe the basic theory behind these methods and we show three of their successful applications to solving machine learning problems: regularized risk minimization, multiple kernel learning, and MAP inference in graphical models.

Many problems in machine learning are elegantly translated to convex optimization problems, which, however, are sometimes difficult to solve efficiently by off-the-shelf solvers. This difficulty can stem from complexity of either the feasible set or of the objective function. Often, these can be accessed only indirectly via an oracle. To access a feasible set, the oracle either asserts that a given query point lies in the set or finds a hyperplane

that separates the point from the set. To access an objective function, the oracle returns the value and a subgradient of the function at the query point. Cutting plane methods solve the optimization problem by approximating the feasible set or the objective function by a bundle of linear inequalities, called cutting planes. The approximation is iteratively refined by adding new cutting planes, computed from the responses of the oracle.

Cutting plane methods have been extensively studied in literature. We refer to Boyd and Vandenberge (2008) for an introductory yet comprehensive overview. For the sake of self consistency, we review the basic theory in Section 1.1. Then, in three separate sections, we describe their successful applications to three machine learning problems.

The first application, Section 1.2, is on learning linear predictors from data based on *regularized risk minimization* (RRM). RRM often leads to a convex but nonsmooth task, which cannot be efficiently solved by general-purpose algorithms, especially for large-scale data. Prominent examples of RRM are support vector machines, logistic regression, and structured output learning. We review a generic risk minimization algorithm proposed by Teo et al. (2007, 2010), inspired by a variant of cutting plane methods known as proximal bundle methods. We also discuss its accelerated version (Franc and Sonnenburg, 2008, 2010; Teo et al., 2010), which is among the fastest solvers for the large-scale learning.

The second application, Section 1.3, is *multiple kernel learning* (MKL). While classical kernel-based learning algorithms use a single kernel, it is sometimes desirable to use multiple kernels (Lanckriet et al., 2004b). Here, we focus on the convex formulation of the MKL problem for classification as first stated in (Zien and Ong, 2007; Rakotomamonjy et al., 2007). We show how this problem can be efficiently solved by a cutting plane algorithm recycling standard SVM implementations. The resulting MKL solver is equivalent to the column generation approach applied to the semi-infinite programming formulation of the MKL problem proposed by Sonnenburg et al. (2006a).

The third application, Section 1.4, is *maximum a posteriori (MAP) inference in graphical models*. It leads to a combinatorial optimization problem which can be formulated as a linear optimization over the marginal polytope (Wainwright and Jordan, 2008). Cutting plane methods iteratively construct a sequence of progressively tighter outer bounds of the marginal polytope, corresponding to a sequence of LP relaxations. We revisit the approach by Werner (2008a, 2010), in which a dual cutting plane method is a straightforward extension of a simple message passing algorithm. It is a generalization of the dual LP relaxation approach by Shlezinger (1976) and the max-sum diffusion algorithm by Kovalevsky and Koval (approx. 1975).

1.1 Introduction to cutting plane methods

Suppose we want to solve the optimization problem

$$\min\{\, f(\boldsymbol{x}) \mid \boldsymbol{x} \in X \,\}\,, \tag{1}$$

where $X \subseteq \mathbb{R}^n$ is a convex set, $f\colon \mathbb{R}^n \to \mathbb{R}$ is a convex function, and we assume that the minimum exists. Set X can be accessed only via the so called *separation oracle* (or *separation algorithm*). Given $\hat{\boldsymbol{x}} \in \mathbb{R}^n$, the separation oracle either asserts that $\hat{\boldsymbol{x}} \in X$ or returns a hyperplane $\langle \boldsymbol{a}, \boldsymbol{x} \rangle \leq b$ (called a *cutting plane*) that separates $\hat{\boldsymbol{x}}$ from X, i.e., $\langle \boldsymbol{a}, \hat{\boldsymbol{x}} \rangle > b$ and $\langle \boldsymbol{a}, \boldsymbol{x} \rangle \leq b$ for all $\boldsymbol{x} \in X$. Figure 1.1(a) illustrates the idea.

cutting plane algorithm

The *cutting plane algorithm* (Algorithm 1.1) solves (1) by constructing progressively tighter convex polyhedrons X_t containing the true feasible set X, by cutting off infeasible parts of an initial polyhedron X_0. It stops when $\boldsymbol{x}_t \in X$ (possibly up to some tolerance).

The trick behind the method is not to approximate X well by a convex polyhedron but to do so *only near the optimum*. This is best seen if X is already a convex polyhedron, described by a set of linear inequalities. At optimum, only some of the inequalities are active. We could in fact remove all the inactive inequalities without affecting the problem. Of course, we do not know which ones to remove until we know the optimum. The cutting plane algorithm imposes more than the minimal set of inequalities but still possibly much fewer than the whole original description of X.

Algorithm 1.1 Cutting plane algorithm

1: **Initialization:** $t \leftarrow 0$, $X_0 \supseteq X$
2: **loop**
3: Let $\boldsymbol{x}_t \in \operatorname{argmin}_{\boldsymbol{x} \in X_t} f(\boldsymbol{x})$
4: If $\boldsymbol{x}_t \in X$ then stop, else find a cutting plane $\langle \boldsymbol{a}, \boldsymbol{x} \rangle \leq b$ separating $\boldsymbol{x}_t$ from X.
5: $X_{t+1} \leftarrow X_t \cap \{\, \boldsymbol{x} \mid \langle \boldsymbol{a}, \boldsymbol{x} \rangle \leq b \,\}$
6: $t \leftarrow t + 1$
7: **end loop**

This basic idea has many incarnations. Next we describe three of them, which have been used in the three machine learning applications presented in this chapter. Section 1.1.1 describes a cutting plane method suited for minimization of nonsmooth convex functions. An improved variant thereof, called the *bundle method*, is described in Section 1.1.2. Finally, Section 1.1.3 describes application of cutting plane methods to solving combinatorial optimization problems.

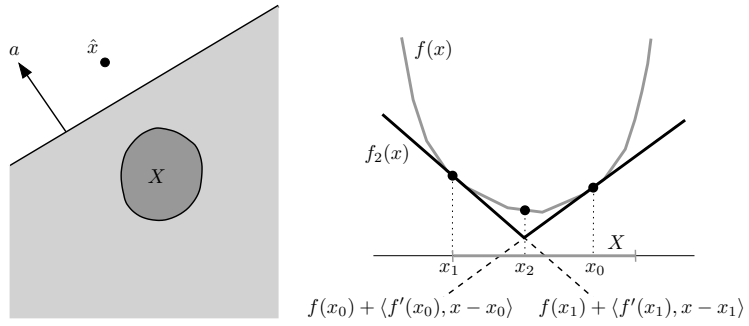


Figure 1.1: Figure (a) illustrates the cutting plane $\langle \boldsymbol{a}, \boldsymbol{x} \rangle \leq b$ cutting off the query point $\hat{\boldsymbol{x}}$ from the light gray halfspace $\{\boldsymbol{x} \mid \langle \boldsymbol{a}, \boldsymbol{x} \rangle \leq b\}$ which contains the feasible set X (dark gray). Figure (b) shows a feasible set X (gray interval) and a function $f(\boldsymbol{x})$ which is approximated by a cutting plane model $f_2(\boldsymbol{x}) = \max\{f(\boldsymbol{x}_0) + \langle f'(\boldsymbol{x}_0), \boldsymbol{x} - \boldsymbol{x}_0 \rangle, f(\boldsymbol{x}_1) + \langle f'(\boldsymbol{x}_1), \boldsymbol{x} - \boldsymbol{x}_1 \rangle\}$. Starting from $\boldsymbol{x}_0$, the CPA generates points $\boldsymbol{x}_1$ and $\boldsymbol{x}_2 = \operatorname{argmin}_{\boldsymbol{x} \in X} f_2(\boldsymbol{x})$.

1.1.1 Nonsmooth optimization

When f is a complicated nonsmooth function while the set X is simple, we want to avoid explicit minimization of f in the algorithm. This can be done by writing (1) in the epigraph form as

$$\min\{\, y \mid (\boldsymbol{x}, y) \in Z \,\} \quad \text{where} \quad Z = \{\, (\boldsymbol{x}, y) \in X \times \mathbb{R} \mid f(\boldsymbol{x}) \leq y \,\}\,. \quad (2)$$

subgradient

In this case, cutting planes can be generated by means of subgradients. Recall that $f'(\hat{\boldsymbol{x}}) \in \mathbb{R}^n$ is a subgradient of f at $\hat{\boldsymbol{x}}$ if

$$f(\boldsymbol{x}) \geq f(\hat{\boldsymbol{x}}) + \langle f'(\hat{\boldsymbol{x}}), \boldsymbol{x} - \hat{\boldsymbol{x}} \rangle\,, \qquad \boldsymbol{x} \in X\,. \quad (3)$$

Thus, the right-hand side is a linear underestimator of f. Assume that $\hat{\boldsymbol{x}} \in X$. Then, the separation algorithm for the set Z can be constructed as follows. If $f(\hat{\boldsymbol{x}}) \leq \hat{y}$ then $(\hat{\boldsymbol{x}}, \hat{y}) \in Z$. If $f(\hat{\boldsymbol{x}}) > \hat{y}$ then the inequality

$$y \geq f(\hat{\boldsymbol{x}}) + \langle f'(\hat{\boldsymbol{x}}), \boldsymbol{x} - \hat{\boldsymbol{x}} \rangle \quad (4)$$

defines a cutting plane separating $(\hat{\boldsymbol{x}}, \hat{y})$ from Z.

This leads to the algorithm proposed independently by Cheney and Goldstein (1959) and Kelley (1960). Starting with $\boldsymbol{x}_0 \in X$, it computes the next

iterate $\boldsymbol{x}_t$ by solving

$$\begin{aligned}&(\boldsymbol{x}_t, y_t) \in \operatorname*{argmin}_{(\boldsymbol{x},y)\in Z_t} y \qquad \text{where}\\ &Z_t = \left\{ (\boldsymbol{x}, y) \in X \times \mathbb{R} \mid y \geq f(\boldsymbol{x}_i) + \langle f'(\boldsymbol{x}_i),\, \boldsymbol{x} - \boldsymbol{x}_i \rangle, i = 0, \ldots, t-1 \right\}.\end{aligned} \tag{5}$$

Here, Z_t is a polyhedral outer bound of Z defined by X and the cutting planes from previous iterates $\{\boldsymbol{x}_0, \ldots, \boldsymbol{x}_{t-1}\}$. Problem (5) simplifies to

$$\boldsymbol{x}_t \in \operatorname*{argmin}_{\boldsymbol{x}\in X} f_t(\boldsymbol{x}) \quad \text{where} \quad f_t(\boldsymbol{x}) = \max_{i=0,\ldots,t-1} \left[f(\boldsymbol{x}_i) + \langle f'(\boldsymbol{x}_i),\, \boldsymbol{x} - \boldsymbol{x}_i \rangle \right]. \tag{6}$$

Here, f_t is a *cutting-plane model* of f (see Figure 1.1(b)). Note that $(\boldsymbol{x}_t, f_t(\boldsymbol{x}_t))$ solves (5). By (3) and (6), we have that $f(\boldsymbol{x}_i) = f_t(\boldsymbol{x}_i)$ for $i = 0, \ldots, t-1$ and $f(\boldsymbol{x}) \geq f_t(\boldsymbol{x})$ for $\boldsymbol{x} \in X$, i.e., f_t is an underestimator of f which touches f at the points $\{\boldsymbol{x}_0, \ldots, \boldsymbol{x}_{t-1}\}$. By solving (6) we do not only get an estimate $\boldsymbol{x}_t$ of the optimal point $\boldsymbol{x}^*$ but also a lower bound $f_t(\boldsymbol{x}_t)$ on the optimal value $f(\boldsymbol{x}^*)$. It is natural to terminate when $f(\boldsymbol{x}_t) - f_t(\boldsymbol{x}_t) \leq \varepsilon$, which guarantees that $f(\boldsymbol{x}_t) \leq f(\boldsymbol{x}^*) + \varepsilon$. The method is summarized in Algorithm 1.2.

Algorithm 1.2 Cutting plane algorithm in epigraph form

1: **Initialization:** $t \leftarrow 0$, $\boldsymbol{x}_0 \in X$, $\varepsilon > 0$
2: **repeat**
3: $\quad t \leftarrow t + 1$
4: $\quad$ Compute $f(\boldsymbol{x}_{t-1})$ and $f'(\boldsymbol{x}_{t-1})$.
5: $\quad$ Update the cutting plane model $f_t(\boldsymbol{x}) \leftarrow \max_{i=0,\ldots,t-1} \left[f(\boldsymbol{x}_i) + \langle f'(\boldsymbol{x}_i),\, \boldsymbol{x} - \boldsymbol{x}_i \rangle \right]$
6: $\quad$ Let $\boldsymbol{x}_t \in \operatorname{argmin}_{\boldsymbol{x}\in X} f_t(\boldsymbol{x})$.
7: **until** $f(\boldsymbol{x}_t) - f_t(\boldsymbol{x}_t) \leq \varepsilon$

In Section 1.3, this algorithm is applied to *multiple kernel learning*. This requires solving the problem

$$\min\{\, f(\boldsymbol{x}) \mid \boldsymbol{x} \in X \,\} \qquad \text{where} \qquad f(\boldsymbol{x}) = \max\{\, g(\boldsymbol{\alpha}, \boldsymbol{x}) \mid \boldsymbol{\alpha} \in A \,\}. \tag{7}$$

X is a simplex and function g is linear in $\boldsymbol{x}$ and quadratic negative semi-definite in $\boldsymbol{\alpha}$. In this case, the subgradient $f'(\boldsymbol{x})$ equals the gradient $\nabla_{\boldsymbol{x}} g(\hat{\boldsymbol{\alpha}}, \boldsymbol{x})$ where $\hat{\boldsymbol{\alpha}}$ is obtained by solving a convex quadratic program $\hat{\boldsymbol{\alpha}} \in \operatorname{argmax}_{\boldsymbol{\alpha}\in A} g(\boldsymbol{\alpha}, \boldsymbol{x})$.

1.1.2 Bundle methods

Algorithm 1.2 may converge slowly (Nemirovskij and Yudin, 1983) because subsequent solutions can be very distant, exhibiting a zig-zag behavior, thus many cutting planes do not actually contribute to the approximation of f

around the optimum $\boldsymbol{x}^*$. Bundle methods (Kiwiel, 1983; Lemaréchal et al., 1995) try to reduce this behavior by adding a stabilization term to (6). The *proximal bundle methods* compute the new iterate as

proximal bundle methods

$$\boldsymbol{x}_t \in \operatorname*{argmin}_{\boldsymbol{x}\in X}\{\, \nu_t\|\boldsymbol{x}-\boldsymbol{x}_t^+\|_2^2 + f_t(\boldsymbol{x})\,\}\,,$$

where $\boldsymbol{x}_t^+$ is a current prox-center selected from $\{\boldsymbol{x}_0,\ldots,\boldsymbol{x}_{t-1}\}$ and ν_t is a current stabilization parameter. The added quadratic term ensures that the subsequent solutions are within a ball centered at $\boldsymbol{x}_t^+$ whose radius depends on ν_t. If $f(\boldsymbol{x}_t)$ sufficiently decreases the objective, the *decrease step* is performed by moving the prox-center as $\boldsymbol{x}_{t+1}^+ := \boldsymbol{x}_t$. Otherwise, the *null step* is performed, $\boldsymbol{x}_{t+1}^+ := \boldsymbol{x}_t^+$. If there is an efficient line-search algorithm, the decrease step computes the new prox-center $\boldsymbol{x}_{t+1}^+$ by minimizing f along the line starting at $\boldsymbol{x}_t^+$ and passing through $\boldsymbol{x}_t$. Though bundle methods may improve the convergence significantly they require two parameters: the stabilization parameter ν_t and the minimal decrease in the objective which defines the null step. Despite significantly influencing the convergence, there is no versatile method for choosing these parameters optimally.

In Section 1.2, a variant of this method is applied to *regularized risk minimization* which requires minimizing $f(\boldsymbol{x}) = g(\boldsymbol{x}) + h(\boldsymbol{x})$ over $\mathbb{R}^n$ where g is a simple (typically differentiable) function and h is a complicated nonsmooth function. In this case, the difficulties with setting two parameters are avoided because g naturally plays the role of the stabilization term.

1.1.3 Combinatorial optimization

A typical combinatorial optimization problem can be formulated as

$$\min\{\, \langle \boldsymbol{c}, \boldsymbol{x}\rangle \mid \boldsymbol{x} \in C\,\}\,, \tag{8}$$

where $C \subseteq \mathbb{Z}^n$ (often just $C \subseteq \{0,1\}^n$) is a finite set of feasible configurations, and $\boldsymbol{c} \in \mathbb{R}^n$ is a cost vector. Usually C is combinatorially large but highly structured. Consider the problem

$$\min\{\, \langle \boldsymbol{c}, \boldsymbol{x}\rangle \mid \boldsymbol{x} \in X\,\} \quad \text{where} \quad X = \operatorname{conv} C\,. \tag{9}$$

Clearly, X is a polytope (bounded convex polyhedron) with integral vertices. Hence, (9) is a linear program. Since a solution of a linear program is always attained at a vertex, problems (8) and (9) have the same optimal value. The set X is called the *integral hull* of problem (8).

Integral hulls of hard problems are complex. If a problem (8) is not polynomially solvable then inevitably the number of facets of X is not polynomial. Therefore (9) cannot be solved explicitly. This is where Algorithm 1.1 is

used. The initial polyhedron $X_0 \supseteq X$ is described by a tractable number of linear inequalities and usually it is already a good approximation of X, often but not necessarily we also have $X_0 \cap \mathbb{Z}^n = C$. The cutting plane algorithm then constructs a sequence of gradually tighter LP relaxations of (8).

A fundamental result states that a linear optimization problem and the corresponding separation problem are polynomial-time equivalent (Grötschel et al., 1981). Therefore, for an intractable problem (8) there is no hope to find a polynomial algorithm to separate an arbitrary point from X. However, a polynomial separation algorithm may exist for a *subclass* (even intractably large) of linear inequalities describing X.

After this approach was first proposed by Dantzig et al. (1954) for the travelling salesman problem, it became a breakthrough in tackling hard combinatorial optimization problems. Since then much effort has been devoted to finding good initial LP relaxations X_0 for many such problems, subclasses of inequalities describing integral hulls for these problems, and polynomial separation algorithms for these subclasses. This is the subject of *polyhedral combinatorics* (e.g., Schrijver, 2003).

In Section 1.4, we focus on the NP-hard combinatorial optimization problem arising in MAP inference in graphical models. This problem, in its full generality, has not been properly addressed by the optimization community. We show how its LP relaxation can be incrementally tightened during a message passing algorithm. Because message passing algorithms are dual, this can be understood as a *dual* cutting plane algorithm: it does not add constraints in the primal but variables in the dual. The sequence of approximations of the integral hull X (the marginal polytope) can be seen as arising from lifting and projection.

1.2 Regularized risk minimization

Learning predictors from data is a standard machine learning problem. A wide range of such problems are special instances of the *regularized risk minimization*. In this case, learning is often formulated as an unconstrained minimization of a convex function

$$\boldsymbol{w}^* \in \operatorname*{argmin}_{\boldsymbol{w} \in \mathbb{R}^n} F(\boldsymbol{w}) \qquad \text{where} \qquad F(\boldsymbol{w}) = \lambda \Omega(\boldsymbol{w}) + R(\boldsymbol{w}) \,. \tag{10}$$

The objective $F\colon \mathbb{R}^n \to \mathbb{R}$, called *regularized risk*, is composed of a regularization term $\Omega\colon \mathbb{R}^n \to \mathbb{R}$ and empirical risk $R\colon \mathbb{R}^n \to \mathbb{R}$ which are both convex functions. The number $\lambda \in \mathbb{R}_+$ is a predefined regularization constant and $\boldsymbol{w} \in \mathbb{R}^n$ is a parameter vector to be learned. The regularization term Ω

is typically a simple, cheap-to-compute function used to constrain the space of solutions in order to improve generalization. The empirical risk R evaluates how well the parameters $\boldsymbol{w}$ explains the training examples. Evaluation of R is often computationally expensive.

Example 1.1. *Given a set of training examples* $\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_m, y_m)\} \in (\mathbb{R}^n \times \{+1, -1\})^m$*, the goal is to learn a parameter vector* $\boldsymbol{w} \in \mathbb{R}^n$ *of a linear classifier* $h\colon \mathbb{R}^n \to \{-1, +1\}$ *which returns* $h(\boldsymbol{x}) = +1$ *if* $\langle \boldsymbol{x}, \boldsymbol{w} \rangle \geq 0$ *and* $h(\boldsymbol{x}) = -1$ *otherwise. Linear support vector machines (Cortes and Vapnik, 1995) without bias learn the parameter vector* $\boldsymbol{w}$ *by solving (10) with the regularization term* $\Omega(\boldsymbol{w}) = \frac{1}{2}\|\boldsymbol{w}\|_2^2$ *and the empirical risk* $R(\boldsymbol{w}) = \frac{1}{m}\sum_{i=1}^{m} \max\{0, 1 - y_i \langle \boldsymbol{x}_i, \boldsymbol{w} \rangle\}$ *which, in this case, is a convex upper bound on the number of mistakes the classifier* $h(\boldsymbol{x})$ *makes on the training examples.*

There is a long list of learning algorithms which in their core are solvers of a special instance of (10), see, e.g. Schölkopf and Smola (2002). If F is differentiable, (10) is solved by algorithms for a smooth optimization. If F is nonsmooth, (10) is typically transformed to an equivalent problem solvable by off-the-shelf methods. For example, learning of the linear SVM classifier in Example 1.1 can be equivalently expressed as quadratic program. Because off-the-shelf solvers are often not efficient enough in practice a huge effort has been put into development of specialized algorithms tailored to particular instances of (10).

Teo et al. (2007, 2010) proposed a generic algorithm to solve (10) which is a modification of the proximal bundle methods. The algorithm, called *bundle method for risk minimization* (BMRM), exploits the specific structure of the objective F in (10). In particular, only the risk term R is approximated by the cutting-plane model while the regularization term Ω is without any change used to stabilize the optimization. In contrast, standard bundle methods introduce the stabilization term artificially. The resulting BMRM is highly modular and was proven to converge in $\mathcal{O}(\frac{1}{\varepsilon})$ iterations to an ε-precise solution. In addition, if an efficient line-search algorithm is available, BMRM can be drastically accelerated with a technique proposed by Franc and Sonnenburg (2008, 2010); Teo et al. (2010). The accelerated BMRM has been shown to be highly competitive with state-of-the-art solvers tailored to particular instances of (10).

In the next two sections, we describe BMRM algorithm and its version accelerated by line-search.

Algorithm 1.3 Bundle Method for Regularized Risk Minimization (BMRM)

input & initialization: $\varepsilon > 0$, $\boldsymbol{w}_0 \in \mathbb{R}^n$, $t \leftarrow 0$
repeat
  $t \leftarrow t + 1$
  Compute $R(\boldsymbol{w}_{t-1})$ and $R'(\boldsymbol{w}_{t-1})$
  Update the model $R_t(\boldsymbol{w}) \leftarrow \max_{i=0,\ldots,t-1} R(\boldsymbol{w}_i) + \langle R'(\boldsymbol{w}_i), \boldsymbol{w} - \boldsymbol{w}_i \rangle$
  Solve the reduced problem $\boldsymbol{w}_t \leftarrow \operatorname{argmin}_{\boldsymbol{w}} F_t(\boldsymbol{w})$ where $F_t(\boldsymbol{w}) = \lambda\Omega(\boldsymbol{w}) + R_t(\boldsymbol{w})$
until $F(\boldsymbol{w}_t) - F_t(\boldsymbol{w}_t) \leq \varepsilon$

1.2.1 Bundle method for regularized risk minimization

Following optimization terminology, we will call (10) the *master problem*. Using the approach by Teo et al. (2007), one can approximate the master problem (10) by its *reduced problem*

$$\boldsymbol{w}_t \in \operatorname*{argmin}_{\boldsymbol{w} \in \mathbb{R}^n} F_t(\boldsymbol{w}) \qquad \text{where} \qquad F_t(\boldsymbol{w}) = \lambda\Omega(\boldsymbol{w}) + R_t(\boldsymbol{w})\,. \tag{11}$$

The reduced problem (11) is obtained from the master problem (10) by substituting the cutting-plane model R_t for the empirical risk R while the regularization term Ω remains unchanged. The cutting-plane model reads

$$R_t(\boldsymbol{w}) = \max_{i=0,\ldots,t-1} \left[R(\boldsymbol{w}_i) + \langle R'(\boldsymbol{w}_i),\, \boldsymbol{w} - \boldsymbol{w}_i \rangle \right], \tag{12}$$

where $R'(\boldsymbol{w}) \in \mathbb{R}^n$ is a subgradient of R at point $\boldsymbol{w}$. Since $R(\boldsymbol{w}) \geq R_t(\boldsymbol{w})$, $\forall \boldsymbol{w} \in \mathbb{R}^n$, the reduced problem's objective F_t is an underestimator of the master objective F. Starting from $\boldsymbol{w}_0 \in \mathbb{R}^n$, BMRM of Teo et al. (2007) (Algorithm 1.3) computes a new iterate $\boldsymbol{w}_t$ by solving the reduced problem (11). In each iteration t, the cutting-plane model (12) is updated by a new cutting plane computed at the intermediate solution $\boldsymbol{w}_t$ leading to a progressively tighter approximation of F. The algorithm halts if the gap between the upper bound $F(\boldsymbol{w}_t)$ and the lower bound $F_t(\boldsymbol{w}_t)$ falls bellow a desired ε, meaning that $F(\boldsymbol{w}_t) \leq F(\boldsymbol{w}^*) + \varepsilon$.

Solving the reduced problem

In practice, the number of cutting planes t required before the algorithm converges is typically much lower than the dimension n of the parameter vector $\boldsymbol{w} \in \mathbb{R}^n$. Thus, it is beneficial to solve the reduced problem (11) in its dual formulation. Let $A = [\boldsymbol{a}_0, \ldots, \boldsymbol{a}_{t-1}] \in \mathbb{R}^{n \times t}$ be a matrix whose columns are the subgradients $\boldsymbol{a}_i = R'(\boldsymbol{w}_i)$ and let $\boldsymbol{b} = [b_0, \ldots, b_{t-1}] \in \mathbb{R}^t$ be a column vector whose components equal to $b_i = R(\boldsymbol{w}_i) - \langle R'(\boldsymbol{w}_i), \boldsymbol{w}_i \rangle$. Then the reduced problem (11) can be equivalently expressed as

$$\boldsymbol{w}_t \in \operatorname*{argmin}_{\boldsymbol{w} \in \mathbb{R}^n, \xi \in \mathbb{R}} \left[\lambda\Omega(\boldsymbol{w}) + \xi \right] \quad \text{s.t.} \quad \xi \geq \langle \boldsymbol{w},\, \boldsymbol{a}_i \rangle + b_i\,,\ i = 0, \ldots, t-1\,. \tag{13}$$

The Lagrange dual of (13) reads (Teo et al., 2010, Theorem 2)

$$\boldsymbol{\alpha}_t \in \operatorname*{argmin}_{\boldsymbol{\alpha}\in\mathbb{R}^t} \left[-\lambda\Omega^*(-\lambda^{-1}A\boldsymbol{\alpha}) + \langle \boldsymbol{\alpha}, \boldsymbol{b}\rangle \right] \quad \text{s.t. } \|\boldsymbol{\alpha}\|_1 = 1\,, \boldsymbol{\alpha} \geq 0\,, \tag{14}$$

where $\Omega^*\colon \mathbb{R}^n \to \mathbb{R}^t$ denotes the Fenchel dual of Ω defined as

$$\Omega^*(\boldsymbol{\mu}) = \sup\left\{ \langle \boldsymbol{w}, \boldsymbol{\mu}\rangle - \Omega(\boldsymbol{w}) \;\middle|\; \boldsymbol{w} \in \mathbb{R}^n \right\}.$$

Having the dual solution $\boldsymbol{\alpha}_t$, the primal solution can be computed by solving $\boldsymbol{w}_t \in \operatorname{argmax}_{\boldsymbol{w}\in\mathbb{R}^n} \left[\langle \boldsymbol{w}, -\lambda^{-1}A\boldsymbol{\alpha}_t\rangle - \Omega(\boldsymbol{w})\right]$ which for differentiable Ω simplifies to $\boldsymbol{w}_t = \nabla_{\boldsymbol{\mu}}\Omega^*(-\lambda^{-1}A\boldsymbol{\alpha}_t)$.

Example 1.2. *For the quadratic regularizer $\Omega(\boldsymbol{w}) = \frac{1}{2}\|\boldsymbol{w}\|_2^2$ the Fenchel dual reads $\Omega^*(\boldsymbol{\mu}) = \frac{1}{2}\|\boldsymbol{\mu}\|_2^2$. The dual reduced problem (14) boils down to the quadratic program*

$$\boldsymbol{\alpha}_t \in \operatorname*{argmin}_{\boldsymbol{\alpha}\in\mathbb{R}^t} \left[-\frac{1}{2\lambda}\boldsymbol{\alpha}^T A^T A\boldsymbol{\alpha} + \boldsymbol{\alpha}^T\boldsymbol{b} \right] \quad \textit{s.t.} \quad \|\boldsymbol{\alpha}\|_1 = 1\,, \boldsymbol{\alpha} \geq 0$$

and the primal solution can be computed analytically by $\boldsymbol{w}_t = -\lambda^{-1}A\boldsymbol{\alpha}_t$.

Convergence guarantees

The convergence of Algorithm 1.3 in a finite number of iterations is guaranteed by the following theorem:

Theorem 1.3. *(Teo et al., 2010, Theorem 5) Assume that (i) $F(\boldsymbol{w}) \geq 0$, $\forall \boldsymbol{w} \in \mathbb{R}^n$, (ii) $\max_{\boldsymbol{g}\in\partial R(\boldsymbol{w})} \|\boldsymbol{g}\|_2 \leq G$ for all $\boldsymbol{w} \in \{\boldsymbol{w}_0, \ldots, \boldsymbol{w}_{t-1}\}$ where $\partial R(\boldsymbol{w})$ denotes the subdifferential of R at point $\boldsymbol{w}$, and (iii) Ω^* is twice differentiable and has bounded curvature, that is, $\|\partial^2\Omega^*(\boldsymbol{\mu})\| \leq H^*$ for all $\boldsymbol{\mu} \in \{\, \boldsymbol{\mu}' \in \mathbb{R}^t \mid \boldsymbol{\mu}' = \lambda^{-1}A\boldsymbol{\alpha}\,, \|\boldsymbol{\alpha}\|_1 = 1\,, \boldsymbol{\alpha} \geq 0\,\}$ where $\partial^2\Omega^*(\boldsymbol{\mu})$ is the Hessian of Ω^* at point $\boldsymbol{\mu}$. Then Algorithm 1.3 terminates after at most*

$$T \leq \log_2 \frac{\lambda F(0)}{G^2 H^*} + \frac{8G^2H^*}{\lambda\varepsilon} - 1$$

iterations for any $\varepsilon < 4G^2H^*\lambda^{-1}$.

Furthermore, for a twice differentiable F with bounded curvature Algorithm 1.3 requires only $\mathcal{O}(\log\frac{1}{\varepsilon})$ iterations instead of $\mathcal{O}(\frac{1}{\varepsilon})$ (Teo et al., 2010, Theorem 5). The most constraining assumption of Theorem 1.3 is that it requires Ω^* to be twice differentiable. This assumption holds, e.g., for the quadratic $\Omega(\boldsymbol{w}) = \frac{1}{2}\|\boldsymbol{w}\|_2^2$ and the negative entropy $\Omega(\boldsymbol{w}) = \sum_{i=1}^n w_i \log w_i$ regularizers. Unfortunately, the theorem does not apply for the ℓ_1-norm regularizer $\Omega(\boldsymbol{w}) = \|\boldsymbol{w}\|_1$ often used to enforce sparse solutions.

1.2.2 BMRM algorithm accelerated by line-search

BMRM can be drastically accelerated whenever an efficient line-search algorithm for the master objective F is available. An accelerated BMRM for solving linear SVM problem (c.f. Example 1.1) has been first proposed in Franc and Sonnenburg (2008). Franc and Sonnenburg (2010) generalized the method for solving (10) with an arbitrary risk R and quadratic regularizer $\Omega(\boldsymbol{w}) = \frac{1}{2}\|\boldsymbol{w}\|_2^2$. Finally, Teo et al. (2010) proposed a fully general version imposing no restrictions on Ω and R. BMRM accelerated by the line-search, in Teo et al. (2010) called LS-BMRM, is described by Algorithm 1.4.

Algorithm 1.4 BMRM accelerated by line-search (LS-BMRM)

1: **input & initialization:** $\varepsilon \geq 0$, $\theta \in (0, 1]$, $\boldsymbol{w}_0^b$, $\boldsymbol{w}_0^c \leftarrow \boldsymbol{w}_0^b$, $t \leftarrow 0$
2: **repeat**
3: $\quad t \leftarrow t + 1$
4: $\quad$ Compute $R(\boldsymbol{w}_{t-1}^c)$ and $R'(\boldsymbol{w}_{t-1}^c)$
5: $\quad$ Update the model $R_t(\boldsymbol{w}) \leftarrow \max_{i=1,\ldots,t-1} R(\boldsymbol{w}_i^c) + \langle R'(\boldsymbol{w}_i^c), \boldsymbol{w} - \boldsymbol{w}_i^c \rangle$
6: $\quad \boldsymbol{w}_t \leftarrow \operatorname{argmin}_{\boldsymbol{w}} F_t(\boldsymbol{w})$ where $F_t(\boldsymbol{w}) = \lambda\Omega(\boldsymbol{w}) + R_t(\boldsymbol{w})$
7: $\quad$ Line-search: $k_t \leftarrow \operatorname{argmin}_{k \geq 0} F(\boldsymbol{w}_t^b + k(\boldsymbol{w}_t - \boldsymbol{w}_{t-1}^b))$
8: $\quad \boldsymbol{w}_t^b \leftarrow \boldsymbol{w}_{t-1}^b + k_t(\boldsymbol{w}_t - \boldsymbol{w}_{t-1}^b)$
9: $\quad \boldsymbol{w}_t^c \leftarrow (1 - \theta)\boldsymbol{w}_{t-1}^b + \theta\boldsymbol{w}_t$
10: **until** $F(\boldsymbol{w}_t^b) - F_t(\boldsymbol{w}_t) \leq \varepsilon$

Unlike BMRM, LS-BMRM simultaneously optimizes the master and reduced problems' objectives F and F_t, respectively. In addition, LS-BMRM selects cutting planes that are close to the best-so-far solution which has a stabilization effect and, moreover, such cutting planes have a higher chance of actively contributing to the approximation of the master objective F around the optimum $\boldsymbol{w}^*$. In particular, there are three main changes compared to BMRM:

1. BMRM-LS maintains the best-so-far solution $\boldsymbol{w}_t^b$ obtained during the first t iterations, i.e., $F(\boldsymbol{w}_0^b), \ldots, F(\boldsymbol{w}_t^b)$ is a monotonically decreasing sequence.

2. The new best-so-far solution $\boldsymbol{w}_t^b$ is found by searching along a line starting at the previous solution $\boldsymbol{w}_{t-1}^b$ and crossing the reduced problem's solution $\boldsymbol{w}_t$. This is implemented on lines 7 and 8.

3. The new cutting plane is computed to approximate the master objective F at the point $\boldsymbol{w}_t^c \leftarrow (1 - \theta)\boldsymbol{w}_t^b + \theta\boldsymbol{w}_t$ (line 9) which lies on the line segment between the best-so-far solution $\boldsymbol{w}_t^b$ and the reduced problem's solution $\boldsymbol{w}_t$. $\theta \in (0, 1]$ is a prescribed parameter. Note that $\boldsymbol{w}_t^c$ must not be set directly to $\boldsymbol{w}_t^b$ in order to guarantee convergence (i.e., $\theta = 0$ is not allowed). It was found experimentally (Franc and Sonnenburg, 2010), that value $\theta = 0.1$

works consistently well.

Convergence guarantees

LS-BMRM converges in $\mathcal{O}(\frac{1}{\varepsilon})$ iterations to ε-precise solution:

Theorem 1.4. *(Teo et al., 2010, Theorem 7) Under the assumption of Theorem 1.3 Algorithm 1.4 converges to the desired precision after*

$$T \leq \frac{8G^2H^*}{\lambda\varepsilon}$$

iterations for any $\varepsilon < 4G^2H^*\lambda^{-1}$.

Efficient line-search algorithm

LS-BMRM requires at line 7 to solve a line-search problem

$$k^* = \operatorname*{argmin}_{k\geq 0} f(k) \quad \text{where} \quad f(k) = \lambda\Omega(\boldsymbol{w}' + k\boldsymbol{w}) + R(\boldsymbol{w}' + k\boldsymbol{w})\,. \quad (15)$$

Franc and Sonnenburg (2008, 2010) proposed a line-search algorithm which finds the exact solution of (15) if $\Omega(\boldsymbol{w}) = \frac{1}{2}\|\boldsymbol{w}\|_2^2$ and

$$R(\boldsymbol{w}) = \sum_{i=1}^{m} \max_{j=1,\ldots,p} \left(u_{ij} + \langle \boldsymbol{v}_{ij},\, \boldsymbol{w}\rangle\right), \quad (16)$$

where $u_{ij} \in \mathbb{R}$ and $\boldsymbol{v}_{ij} \in \mathbb{R}^n$, $i = 1,\ldots,m$, $j = 1,\ldots,p$, are some fixed scalars and vectors, respectively. In this case, the subdifferential of $\partial f(k)$ can be described by $\mathcal{O}(pm)$ line segments in 2D. Problem (15) can be replaced by solving $\partial f(k) \ni 0$ w.r.t. k which is equivalent to finding among the line segments the one intersecting the x-axis. This line-search algorithm finds the exact solution of (15) in $\mathcal{O}(mp^2 + mp\log mp)$ time. The risk (16) emerges in most variants of the support vector machines learning algorithms, e.g., binary SVMs, multi-class SVMs or SVM regression. Unfortunately, the algorithm is not applicable if p is huge which excludes applications to the structured output SVM learning (Tsochantaridis et al., 2005).

1.2.3 Conclusions

A notable advantage of BMRM is its modularity and simplicity. One only needs to supply a procedure to compute the risk $R(\boldsymbol{w})$ and its subgradient $R'(\boldsymbol{w})$ at a point $\boldsymbol{w}$. The core part of BMRM, i.e., solving the reduced problem, remains for a given regularizer Ω unchanged. Thus, many existing learning problems can be solved by a single optimization technique. Moreover, one can easily experiment with new learning formulations just by specifying the risk term R and its subgradient R' without spending time on development of a new solver for that particular problem.

The convergence speed of BMRM and the accelerated LS-BMRM has

been extensively studied on a variety of real-life problems in domains ranging from the text classification, bioinformatics and computer vision to computer security systems (Teo et al., 2007; Franc and Sonnenburg, 2008, 2010; Teo et al., 2010). Compared to the state-of-the-art dedicated solvers, BMRM is typically slightly slower, however, it is still competitive and practically useful. On the other hand, the LS-BMRM has proved to be among the fastest optimization algorithms for a variety of problems. Despite the similar theoretical convergence times, in practice, the LS-BMRM is on average by an order of magnitude faster than BMRM.

The most time-consuming part of BMRM, as well as LS-BMRM, is the evaluation of the risk R and its subgradient R'. Fortunately, the risk, and thus also its subgradient, are typically additively decomposable which allows for an efficient parallelization of their computation. The effect of the parallelization on the reduction of the computational time is empirically studied in Franc and Sonnenburg (2010); Teo et al. (2010).

A relatively high memory requirements of BMRM/LS-BMRM may be the major deficiency if the method is applied to large-scale problems. The method stores in each iteration t a cutting plane of size $\mathcal{O}(n)$, where n is the dimension of the parameter vector $\boldsymbol{w} \in \mathbb{R}^n$, which leads to $\mathcal{O}(nt)$ memory complexity not counting the reduced problem which is typically much less memory demanding. To alleviate the problem, Teo et al. (2010) propose a limited memory variant of BMRM maintaining up to K cutting planes aggregated from the original t cutting planes. Though the memory limited variant does not have an impact on the theoretical upper bound of the number of iterations, in practice, it may significantly slow down the convergence.

The implementations of BMRM and LS-BMRM can be found in the SHOGUN machine learning toolbox (Sonnenburg et al., 2010) or in the open-source packages BMRM (`http://users.cecs.anu.edu.au/~chteo/BMRM.html`) and LIBOCAS (`http://cmp.felk.cvut.cz/~xfrancv/ocas/html/index.html`).

1.3 Multiple kernel learning

Multiple kernel learning (MKL) (e.g., Bach et al., 2004) has recently become an active line of research. Given a mapping $\Phi : \mathcal{X} \mapsto \mathbb{R}^n$ that represents each object $\boldsymbol{x} \in \mathcal{X}$ in n-dimensional feature space[1], a kernel machine employs a

1. For the sake of simplicity, we consider the n-dimensional Euclidean feature space. However, all the methods in this section can be applied even if the objects are mapped

kernel function

$$\mathbf{k}(\boldsymbol{x}, \boldsymbol{x}') = \langle \Phi(\boldsymbol{x}), \Phi(\boldsymbol{x}') \rangle$$

to compare two objects $\boldsymbol{x}$ and $\boldsymbol{x}'$ without ever *explicitly* computing $\Phi(\boldsymbol{x})$. Ultimately, a kernel machine learns $\boldsymbol{\alpha}$-weighted linear combination of kernel functions with bias b

$$h(\boldsymbol{x}) = \sum_{i=1}^{m} \alpha_i \mathbf{k}(\boldsymbol{x}_i, \boldsymbol{x}) + b\,, \tag{17}$$

where $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m$ is a set of training objects. For example, the support vector machine (SVM) classifier uses the sign of $h(\boldsymbol{x})$ to assign a class label $y \in \{-1, +1\}$ to the object $\boldsymbol{x}$ (e.g., Schölkopf and Smola, 2002).

Traditionally, just a single kernel function has been used. However, it has proven beneficial to consider not just a single, but multiple kernels in various applications (see Section 1.3.4). Currently, the most popular way to combine kernels is via convex combinations, i.e., introducing

$$B = \left\{ \boldsymbol{\beta} \in \mathbb{R}^K \,\middle|\, \|\boldsymbol{\beta}\|_1 = 1\,, \boldsymbol{\beta} \geq 0 \right\}, \tag{18}$$

the *composite kernel* is defined as

$$\mathbf{k}(\boldsymbol{x}, \boldsymbol{x}') = \sum_{k=1}^{K} \beta_k \mathbf{k}_k(\boldsymbol{x}, \boldsymbol{x}')\,, \qquad \boldsymbol{\beta} \in B\,, \tag{19}$$

where $\mathbf{k}_k \colon \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, $k = 1, \ldots, K$, is a given set of positive-definite kernels (Schölkopf and Smola, 2002). Now, in contrast to single kernel algorithms, MKL learns in addition to the coefficients $\boldsymbol{\alpha}$ and b the weighting over kernels $\boldsymbol{\beta}$.

In Section 1.3.1, we review convex MKL for classification and, in Section 1.3.2, we show that this problem can be cast as minimization of a complicated convex function over a simple feasible set. In Section 1.3.3, we derive a CPA that transforms the MKL problem into a sequence of linear and quadratic programs, of which the latter can be efficiently solved by existing SVM solvers. Section 1.3.4 concludes this part.

1.3.1 Convex multiple kernel learning

Various MKL formulations have been proposed (Lanckriet et al., 2004b; Bach et al., 2004; Sonnenburg et al., 2006a; Varma and Babu, 2009; Kloft

---

into arbitrary Reproducing Kernel Hilbert Space (Schölkopf and Smola, 2002).

et al., 2009; Bach, 2009; Nath et al., 2009; Cortes et al., 2009). Here we focus solely on the convex optimization problem for classification as first stated by Zien and Ong (2007); Rakotomamonjy et al. (2007). Note that the same authors have shown that the mixed-norm approaches of Bach et al. (2004); Sonnenburg et al. (2006a) are equivalent.

Let $\{(\boldsymbol{x}_1, y_1), \ldots, (\boldsymbol{x}_m, y_m)\} \in (\mathcal{X} \times \{-1, +1\})^m$ be a training set of examples of input $\boldsymbol{x}$ and output y assumed to be i.i.d. from an unknown distribution $p(\boldsymbol{x}, y)$. The input $\boldsymbol{x}$ is translated into a compositional feature vector $(\Phi_1(\boldsymbol{x}); \ldots; \Phi_K(\boldsymbol{x})) \in \mathbb{R}^{n_1+\cdots+n_k}$ that is constructed by a set of K mappings $\Phi_k \colon \mathcal{X} \to \mathbb{R}^{n_k}$, $k = 1, \ldots, K$. The goal is to predict y from an unseen $\boldsymbol{x}$ by using a linear classifier

$$y = \operatorname{sgn}\big(h(\boldsymbol{x})\big) \quad \text{where} \quad h(\boldsymbol{x}) = \sum_{k=1}^{K} \langle \boldsymbol{w}_k,\, \Phi_k(\boldsymbol{x}) \rangle + b\,, \tag{20}$$

its parameters $\boldsymbol{w}_k \in \mathbb{R}^{n_k}$, $k = 1, \ldots, K$, $b \in \mathbb{R}$, are learned from the training examples. Using the definition $\frac{x}{0} = 0$ if $x = 0$ and ∞ otherwise, the parameters of the classifier (20) can be obtained by solving the following convex *primal MKL optimization problem* (Zien and Ong, 2007; Rakotomamonjy et al., 2007)

$$\begin{aligned} \min \quad & \frac{1}{2} \sum_{k=1}^{K} \frac{1}{\beta_k} \|\boldsymbol{w}_k\|_2^2 + C \sum_{i=1}^{m} \xi_i \\ \text{w.r.t.} \quad & \boldsymbol{\beta} \in B\,, \boldsymbol{w} = (\boldsymbol{w}_1, \ldots, \boldsymbol{w}_K) \in \mathbb{R}^{n_1+\cdots+n_K}\,, \boldsymbol{\xi} \in \mathbb{R}^m, b \in \mathbb{R} \\ \text{s.t.} \quad & \xi_i \geq 0 \text{ and } y_i \left(\sum_{k=1}^{K} \langle \boldsymbol{w}_k, \Phi_k(\boldsymbol{x}_i) \rangle + b \right) \geq 1 - \xi_i,\ i = 1, \ldots, m\,. \end{aligned} \tag{21}$$

Analogously to the SVMs, the objective of (21) is composed of two terms. The first (regularization) term constrains the spaces of the parameters $\boldsymbol{w}_k$, $k = 1, \ldots, K$, in order to improve the generalization of the classifier (20). The second term, weighted by a prescribed constant $C > 0$, is an upper bound on the number of mistakes the classifier (20) makes on the training examples. In contrast to SVMs, positive weights $\boldsymbol{\beta}$ with ℓ_1-norm constraint (see (18)) are introduced to enforce block-wise sparsity, i.e., rather few blocks of features Φ_k are selected (have non-zero weight $\boldsymbol{w}_k$). Since $\frac{1}{\beta_k} \gg 1$ for small β_k, non-zero components of $\boldsymbol{w}_k$ experience stronger penalization and thus the smaller β_k the smoother $\boldsymbol{w}_k$. By definition, $\boldsymbol{w}_k = 0$ if $\beta_k = 0$. Note that for $K = 1$, the MKL problem (21) reduces to the standard two-class linear SVM classifier.

1.3.2 Min-max formulation of multiple kernel learning

To apply kernels, the primal MKL problem (21) must be reformulated such that the features vectors $\Phi_k(\boldsymbol{x}_i)$ appear in terms of dot products only. Following Rakotomamonjy et al. (2007) we can rewrite (21) as

$$\min\{F(\boldsymbol{\beta}) \mid \boldsymbol{\beta} \in B\}\,, \tag{22}$$

where $F(\boldsymbol{\beta})$ is a shortcut for solving the standard SVM primal on the $\boldsymbol{\beta}$-weighted concatenated feature space

$$\begin{aligned} F(\boldsymbol{\beta}) = \min \quad & \frac{1}{2}\sum_{k=1}^{K}\frac{1}{\beta_k}\|\boldsymbol{w}_k\|_2^2 + C\sum_{i=1}^{m}\xi_i \\ \text{w.r.t.} \quad & \boldsymbol{w} = (\boldsymbol{w}_1,\ldots,\boldsymbol{w}_K) \in \mathbb{R}^{n_1+\cdots+n_K}, \boldsymbol{\xi} \in \mathbb{R}^m, b \in \mathbb{R} \\ \text{s.t.} \quad & \xi_i \geq 0 \text{ and } y_i\left(\sum_{k=1}^{K}\langle \boldsymbol{w}_k, \Phi_k(\boldsymbol{x}_i)\rangle + b\right) \geq 1-\xi_i\,, i = 1,\ldots,m. \end{aligned} \tag{23}$$

Note, that in (23) the weights $\boldsymbol{\beta}$ are fixed and the minimization is only over $(\boldsymbol{w}, \boldsymbol{\xi}, b)$. The Lagrange dual of (23) reads (Rakotomamonjy et al., 2007)

$$D(\boldsymbol{\beta}) = \max\{S(\boldsymbol{\alpha}, \boldsymbol{\beta}) \mid \boldsymbol{\alpha} \in A\} \quad \text{where} \quad S(\boldsymbol{\alpha}, \boldsymbol{\beta}) = \sum_{k=1}^{K}\beta_k S_k(\boldsymbol{\alpha})\,, \tag{24}$$

and S_k and A are defined as follows:

$$\begin{aligned} S_k(\boldsymbol{\alpha}) &= \sum_{i=1}^{m}\alpha_i - \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j \langle \Phi_k(\boldsymbol{x}_i),\, \Phi_k(\boldsymbol{x}_j)\rangle \\ A &= \{\boldsymbol{\alpha} \in \mathbb{R}^m \mid 0 \leq \alpha_i \leq C\,, i = 1,\ldots,m\,, \sum_{i=1}^{m}\alpha_i y_i = 0\}\,. \end{aligned} \tag{25}$$

Note, that (24) is equivalent to solving the standard SVM dual with the composite kernel (19). Because (23) is convex and the Slater's qualification condition holds, the duality gap is zero, i.e. $F(\boldsymbol{\beta}) = D(\boldsymbol{\beta})$. Substituting $D(\boldsymbol{\beta})$ for $F(\boldsymbol{\beta})$ in (22) leads to an equivalent *min-max MKL problem*

Min-Max Problem

$$\min\{D(\boldsymbol{\beta}) \mid \boldsymbol{\beta} \in B\}\,. \tag{26}$$

Let $\boldsymbol{\beta}^* \in \operatorname{argmax}_{\boldsymbol{\beta}\in B} D(\boldsymbol{\beta})$ and $\boldsymbol{\alpha}^* \in \operatorname{argmax}_{\boldsymbol{\alpha}\in A} S(\boldsymbol{\alpha}, \boldsymbol{\beta}^*)$. Then the solution of the primal MKL problem (21) can be computed analytically as

$$\boldsymbol{w}_k^* = \beta_k^*\sum_{i=1}^{m}\alpha_i^* y_i \Phi_k(\boldsymbol{x}_i) \quad \text{and} \quad b^* = y_i - \sum_{k=1}^{K}\langle \boldsymbol{w}_k^*,\, \Phi_k(\boldsymbol{x}_i)\rangle\,,\; i \in J\,, \tag{27}$$

where $J = \{j \in \{1, \ldots, m\} \mid 0 < \alpha_i^* < C\}$. The equalities (27) follow from the Karush-Kuhn-Tucker optimality conditions of the problem (23) (e.g., Schölkopf and Smola, 2002). Note, that in practice, b^* is computed as an average over all $|J|$ equalities which is numerically more stable.

By substituting (27) and $\mathbf{k}_k(\boldsymbol{x}_i, \boldsymbol{x}) = \langle \Phi_k(\boldsymbol{x}_i), \Phi_k(\boldsymbol{x}) \rangle$ in the linear classification rule (20), we obtain the kernel classifier (17) with the composite kernel (19). In addition, after substituting $\mathbf{k}_k(\boldsymbol{x}_i, \boldsymbol{x}_j)$ for the dot products $\langle \Phi_k(\boldsymbol{x}_i), \Phi_k(\boldsymbol{x}_j) \rangle$ in (25) we can compute all the parameters of the kernel classifier without ever using the features $\Phi_k(\boldsymbol{x}_i)$ explicitly.

1.3.3 Solving MKL via cutting planes

In this section, we will apply the cutting plane Algorithm 1.2 to the min-max MKL problem (26).

It follows from (24) that the objective D is convex since it is a pointwise maximum over an infinite number of functions $S(\boldsymbol{\alpha}, \boldsymbol{\beta})$, $\boldsymbol{\alpha} \in A$, which are linear in $\boldsymbol{\beta}$ (e.g., Boyd and Vandenberghe, 2004). By Danskin's theorem (e.g., Proposition B.25 in Bertsekas, 1999), the subgradient of D at point $\boldsymbol{\beta}$ equals the gradient $\nabla_{\boldsymbol{\beta}} S(\hat{\boldsymbol{\alpha}}, \boldsymbol{\beta})$ where $\hat{\boldsymbol{\alpha}} \in \operatorname{argmax}_{\boldsymbol{\alpha} \in A} S(\boldsymbol{\alpha}, \boldsymbol{\beta})$, i.e, the subgradient reads

$$D'(\boldsymbol{\beta}) = [S_1(\hat{\boldsymbol{\alpha}}); \ldots; S_K(\hat{\boldsymbol{\alpha}})] \in \mathbb{R}^K \; . \tag{28}$$

Note, that computing $D(\boldsymbol{\beta})$ and its subgradient $D'(\boldsymbol{\beta})$ requires solving the convex quadratic program (24) which is equivalent to the standard SVM dual computed on the composite kernel (19) with a fixed weighting $\boldsymbol{\beta}$ (Rakotomamonjy et al., 2007). Thus, existing SVM solvers are directly applicable.

Having the means to compute D and its subgradient D', we can approximate the objective D by its cutting-plane model

$$\begin{aligned} D_t(\boldsymbol{\beta}) &= \max_{i=0,\ldots,t-1} \left[D(\boldsymbol{\beta}_i) + \langle D'(\boldsymbol{\beta}_i), \boldsymbol{\beta} - \boldsymbol{\beta}_i \rangle \right] \\ &= \max_{i=0,\ldots,t-1} \langle \boldsymbol{\beta}, D'(\boldsymbol{\beta}_i) \rangle \; . \end{aligned} \tag{29}$$

The points $\{\boldsymbol{\beta}_0, \ldots, \boldsymbol{\beta}_{t-1}\}$ can be computed by Kelley's CPA (Algorithm 1.2) as follows. Starting with $\boldsymbol{\beta}_0 \in B$, a new iterate is obtained by solving

$$\boldsymbol{\beta}_t \in \operatorname*{argmin}_{\boldsymbol{\beta} \in B} D_t(\boldsymbol{\beta}) \; , \tag{30}$$

which can be cast as a linear program. Note, that since the feasible set B is bounded so is the solution of (30). In each iteration t, the obtained point $\boldsymbol{\beta}_t$ is an estimate of the optimal $\boldsymbol{\beta}^*$, and it is also used to update the cutting

Algorithm 1.5 Cutting plane algorithm for solving the MKL problem. The algorithm requires solving a simple LP (line 7) and a convex QP (line 3) which is equivalent to the standard SVM dual.

1: **Initialization:** $t \leftarrow 0$, $\boldsymbol{\beta}_0 \in B$ (e.g. $\boldsymbol{\beta}_0 = [\frac{1}{K}; \ldots; \frac{1}{K}]$), $\varepsilon > 0$
2: **repeat**
3: Let $\boldsymbol{\alpha}_t \in \operatorname{argmax}_{\boldsymbol{\alpha} \in A} S(\boldsymbol{\alpha}, \beta_t)$
4: Compute $D(\boldsymbol{\beta}_t) \leftarrow S(\boldsymbol{\alpha}_t, \boldsymbol{\beta}_t)$ and $D'(\boldsymbol{\beta}_t) = [S_1(\boldsymbol{\alpha}_t); \ldots; S_K(\boldsymbol{\alpha}_t)]$
5: $t \leftarrow t + 1$
6: Update the cutting plane model $D_t(\boldsymbol{\beta}) \leftarrow \max_{i=0,\ldots,t-1} \langle D'(\boldsymbol{\beta}_i), \boldsymbol{\beta} \rangle$
7: Let $\boldsymbol{\beta}_t \in \operatorname{argmin}_{\boldsymbol{\beta} \in B} D_t(\boldsymbol{\beta})$
8: **until** $D(\boldsymbol{\beta}_{t-1}) - D_t(\boldsymbol{\beta}_t) \leq \varepsilon$

plane model (29). The process is repeated until the gap between $D(\boldsymbol{\beta}_{t-1})$ and $D_t(\boldsymbol{\beta}_t)$ falls below a prescribed ε, meaning that $D(\boldsymbol{\beta}_t) \leq D(\boldsymbol{\beta}^*) + \varepsilon$ holds. Algorithm 1.5 summarizes the method.

Note that originally Sonnenburg et al. (2006a) converted the problem (26) into a *semi-infinite linear problem* (SILP) that was solved by column generation. However, the SILP is equivalent to the epigraph form of (26) (see Section 1.1.1) and the column generation results in the exact same Algorithm 1.5.

Since large-scale SVM training problems are usually solved by so-called decomposition techniques like chunking (e.g., used in Joachims, 1999), one may significantly speedup Algorithm 1.5 by alternately solving for $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ within the SVM solver avoiding to solve the full SVM model with a high precision (Sonnenburg et al., 2006a). Furthermore, as noted in Section 1.2.1, potential oscilations occuring in cutting plane methods can be reduced by the bundle methods, as has been done by Xu et al. (2009a).

1.3.4 Conclusions

Multiple Kernel Learning has been used in various applications across diverse fields like bioinformatics, image analysis, signal processing, and biomedical applications like brain computer interfaces. It is being applied to fusing heterogeneous data (Lanckriet et al., 2004a; Sonnenburg et al., 2006b; Zien and Ong, 2007; Rakotomamonjy et al., 2008; Varma and Babu, 2009), to understand the learned kernel classifier (Sonnenburg et al., 2005), feature selection (Szafranski et al., 2008; Xu et al., 2009b; Subrahmanya and Shin, 2010) or automated model selection Sonnenburg et al. (2006a). In this section, we have illustrated that the min-max formulation of MKL problem (22) can be converted into a sequence of linear and quadratic programs, of which the LP is of a simple nature and the QP can be directly solved using any of the various existing SVM solvers. There exist further

extensions of this approach not discussed in this section, e.g. an infinite dimensional version of the min-max MKL which was proposed by Argyriou et al. (2006). We provide efficient implementations of MKL in the SHOGUN machine learning toolbox (Sonnenburg et al., 2010).

1.4 MAP inference in graphical models

MAP inference in graphical models (Wainwright and Jordan, 2008) leads to the following NP-hard combinatorial optimization problem: *given a set of variables and a set of functions of (small) subsets of the variables, maximize the sum of the functions over all the variables*. This is also known as the weighted constraint satisfaction problem (Rossi et al., 2006, chapter 9).

The problem has a natural LP relaxation, proposed independently by Shlezinger (1976), Koster et al. (1998), and Wainwright et al. (2005). It is crucial to optimize the LP in the dual because primal methods do not scale to large problems, which is not done in (Koster et al., 1998). The relaxation was extended by Wainwright et al. (2005), Wainwright and Jordan (2008) and Johnson et al. (2007) to a hierarchy of progressively tighter LP relaxations. Komodakis et al. (2007) pointed out that the LP approach can be seen as a dual decomposition of the problem to tractable subproblems.

Several authors proposed to tighten the relaxation incrementally. First, primal methods were proposed (Koster et al., 1998; Sontag and Jaakkola, 2007; Sontag, 2007), then dual methods (Werner, 2008a, 2010; Kumar and Torr, 2008; Sontag et al., 2008; Komodakis and Paragios, 2008). Not all of the authors related these incremental schemes to cutting plane methods.

We revisit here the approach by Werner (2008a, 2010), which, we believe, captures the very core of the dual cutting plane approach to MAP inference in a clean and transparent way. It is a generalization of the dual LP relaxation approach by Shlezinger (1976) and the max-sum diffusion algorithm by Kovalevsky and Koval (approx. 1975), which have been recently reviewed by Werner (2005, 2007).

The approach is surprisingly simple and general. Every subset of the of variables is assigned a function ('interaction'), all of them except a small part (which defines the problem) being initially zero. Max-sum diffusion passes messages between pairs of the variable subsets, acting as reparameterizations of the problem which monotonically decrease its upper bound. While in the extreme case all pairs of variable subsets are coupled like this, coupling only some of them results in a relaxation of the problem. Any time during diffusion we can tighten the relaxation by coupling new pairs – this results in an incremental scheme, recognized as a dual cutting plane method.

After introducing notation, we construct the integer hull of the problem and the hierarchy of its LP relaxations in Section 1.4.2. In Sections 1.4.3 and 1.4.4 we dualize the LP relaxation and describe the max-sum diffusion algorithm which optimizes the dual. In Section 1.4.5 we augment this to a dual cutting plane algorithm and discuss the corresponding separation problem. Section 1.4.6 explains the geometry of this cutting plane algorithm in the primal domain, relating it to the marginal polytope.

1.4.1 Notation and problem definition

Let V be an ordered set of *variables* (the order on V is used only for notation consistency). A variable $v \in V$ attains *states* $x_v \in X_v$, where X_v is the (finite) *domain* of the variable. The *joint domain* of a subset $A \subseteq V$ of the variables is the Cartesian product $X_A = \prod_{v \in A} X_v$, where the order of factors is given by the order on V. A tuple $x_A \in X_A$ is a *joint state* of variables A. An *interaction* with *scope* $A \subseteq V$ is a function $\theta_A \colon X_A \to \overline{\mathbb{R}} = \mathbb{R} \cup \{-\infty\}$.

Let $E \subseteq 2^V$ be a hypergraph on V (a set of subsets of V). Every variable subset $A \subseteq V$ is assigned an interaction, while θ_A is identically zero whenever $A \notin E$. Having to deal with so many interactions may seem scary – but it will always be evident that the vast majority of them do not contribute to sums and are never visited in algorithms. Our task is to compute

$$\max_{x_V \in X_V} \sum_{A \in E} \theta_A(x_A) = \max_{x_V \in X_V} \sum_{A \subseteq V} \theta_A(x_A) \; . \tag{31}$$

E.g., if $V = (1,2,3,4)$ and $E = \{(1,3,4),(2,3),(2,4),(3)\}$ then (31) reads $\max_{x_1,x_2,x_3,x_4} [\theta_{134}(x_1,x_3,x_4) + \theta_{23}(x_2,x_3) + \theta_{24}(x_2,x_4) + \theta_3(x_3)]$. Note, as V is an ordered set we use $(\cdots)$ rather than $\{\cdots\}$ to denote V and its subsets.

We will use $T = \{\, (A, x_A) \mid A \subseteq V,\ x_A \in X_A \,\}$ to denote the set of all joint states of all variable subsets ('T' stands for '$t$uples'). All interactions θ_A, $A \subseteq V$, will be understood as a single vector $\boldsymbol{\theta} \in \overline{\mathbb{R}}^T$.

1.4.2 The hierarchy of LP relaxations

LP formulation

We define a mapping $\boldsymbol{\delta} \colon X_V \to \{0,1\}^T$ as follows: $\delta_A(y_A)(x_V)$ equals 1 if joint state y_A is the restriction of joint state x_V on variables A, and 0 otherwise. Here, $\delta_A(y_A)(x_V)$ denotes the (A, y_A)-component of vector $\boldsymbol{\delta}(x_V) \in \{0,1\}^T$. This lets us write the objective function of (31) as a scalar product,

$$\sum_{A \subseteq V} \theta_A(x_A) = \sum_{A \subseteq V} \sum_{y_A} \theta_A(y_A)\, \delta_A(y_A)(x_V) = \langle \boldsymbol{\theta}, \boldsymbol{\delta}(x_V) \rangle \; .$$

Problem (31) can now be reformulated as

$$\max_{x_V \in X_V} \sum_{A \subseteq V} \theta_A(x_A) = \max_{x_V \in X_V} \langle \boldsymbol{\theta}, \boldsymbol{\delta}(x_V) \rangle = \max_{\boldsymbol{\mu} \in \boldsymbol{\delta}(X_V)} \langle \boldsymbol{\theta}, \boldsymbol{\mu} \rangle = \max_{\boldsymbol{\mu} \in \operatorname{conv} \boldsymbol{\delta}(X_V)} \langle \boldsymbol{\theta}, \boldsymbol{\mu} \rangle$$

where $\boldsymbol{\delta}(X_V) = \{\, \boldsymbol{\delta}(x_V) \mid x_V \in X_V \,\}$. This expresses problem (31) in the form (9), as a linear optimization over the integral hull $\operatorname{conv} \boldsymbol{\delta}(X_V) \subseteq [0,1]^T$.

Let $I = \{\, (A, B) \mid B \subseteq A \subseteq V \,\}$ denote the set of hyperedge pairs related by inclusion, i.e., the inclusion relation on 2^V. For any $J \subseteq I$, we define a polytope $\mathcal{M}(J)$ to be the set of vectors $\boldsymbol{\mu} \in [0,1]^T$ satisfying

$$\sum_{x_{A \setminus B}} \mu_A(x_A) = \mu_B(x_B) \;, \qquad (A, B) \in J,\ x_B \in X_B \;, \tag{32a}$$

$$\sum_{x_A} \mu_A(x_A) = 1 \;, \qquad A \subseteq V \;. \tag{32b}$$

What is this object? Any $\boldsymbol{\mu} \in \mathcal{M}(J)$ is a set of distributions $\mu_A \colon X_A \to [0,1]$ over every subset $A \subseteq V$ of the variables. Constraint (32b) normalizes the distributions. Constraint (32a) couples pairs of distributions, imposing that μ_B is the marginal of μ_A whenever $(A, B) \in J$. E.g., if $A = (1,2,3,4)$ and $B = (2,4)$ then (32a) reads $\sum_{x_1, x_3} \mu_{1234}(x_1, x_2, x_3, x_4) = \mu_{24}(x_2, x_4)$.

integral hull

For brevity, we will use the shorthand $\mathcal{M}(I) = \mathcal{M}$. We claim that

$$\operatorname{conv} \boldsymbol{\delta}(X_V) = \mathcal{M} \;. \tag{33}$$

To see it, let us write a convex combination of the elements of $\boldsymbol{\delta}(X_V)$,

$$\boldsymbol{\mu} = \sum_{x_V} \mu_V(x_V)\, \boldsymbol{\delta}(x_V) \;, \tag{34}$$

where $\mu_V(x_V)$ denote the coefficients of the convex combination. But μ_V is already part of $\boldsymbol{\mu}$. The (A, y_A)-component of vector (34) reads

$$\mu_A(y_A) = \sum_{x_V} \mu_V(x_V)\, \delta_A(y_A)(x_V) = \sum_{y_{V \setminus A}} \mu_V(y_V) \;.$$

But this is (32a) for $(A, B) = (V, A)$.

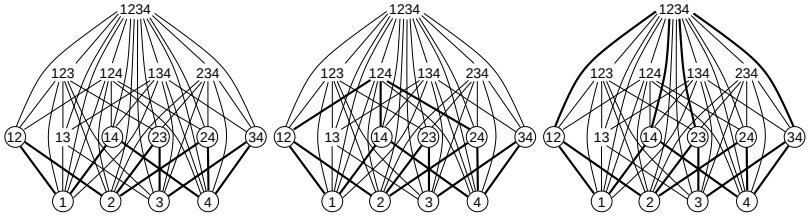
hierarchy of LP relaxations

By imposing only a subset of all possible marginalization constraints (32a), an outer relaxation of the integral hull $\operatorname{conv} \boldsymbol{\delta}(X_V) = \mathcal{M}$ is obtained. Namely, for any $J \subseteq I$ we have $\mathcal{M}(J) \supseteq \mathcal{M}$, hence

$$\max\{\, \langle \boldsymbol{\theta}, \boldsymbol{\mu} \rangle \mid \boldsymbol{\mu} \in \mathcal{M}(J) \,\} \tag{35}$$

is a linear programming relaxation of problem (31), i.e., its optimum is an upper bound on (31). All possible relaxations form a partially ordered hierarchy, indexed by $J \subseteq I$. Figure 1.2 shows examples.

We remark that the hierarchy could be made finer-grained by selecting also *subsets* of joint states, i.e., by imposing marginalization equality (32a) for $(A, B, x_B) \in J$ where $J \subseteq I = \{\, (A, B, x_B) \mid B \subseteq A \subseteq V,\ x_B \in X_B \,\}$.



(a) (b) (c)

Figure 1.2: The Hasse diagram of the set 2^V of all subsets of $V = (1, 2, 3, 4)$. The nodes depict hyperedges $A \subseteq V$ (with hyperedge $\emptyset$ omitted) and the arcs depict hyperedge pairs $(A, B) \in I$. The hyperedges in circles form the problem hypergraph $E = \{(1), (2), (3), (4), (1, 2), (1, 4), (2, 3), (2, 4), (3, 4)\}$, the interactions over the non-circled hyperedges are zero. Any subset $J \subseteq I$ of the arcs yields one possible relaxation (35) of problem (31). Subfigures (a), (b), (c) show three example relaxations, with J depicted as thick arcs.

1.4.3 The dual of the LP relaxation

constructing the dual

Rather than solving the linear program (35) directly, it is much better to solve its dual. This dual is constructed as follows. Let matrices $\boldsymbol{A}$ and $\boldsymbol{B}$ be such that $\boldsymbol{A\mu} = \boldsymbol{0}$ and $\boldsymbol{B\mu} = \boldsymbol{1}$ is the set of equalities (32a) and (32b), respectively. Then (35) can be written as the left linear program below:

$$\langle \boldsymbol{\theta}, \boldsymbol{\mu} \rangle \to \max \qquad \langle \boldsymbol{\psi}, \boldsymbol{1} \rangle \to \min \tag{36a}$$

$$\boldsymbol{A\mu} = \boldsymbol{0} \qquad \boldsymbol{\varphi} \lesseqgtr \boldsymbol{0} \tag{36b}$$

$$\boldsymbol{B\mu} = \boldsymbol{1} \qquad \boldsymbol{\psi} \lesseqgtr \boldsymbol{0} \tag{36c}$$

$$\boldsymbol{\mu} \geq \boldsymbol{0} \qquad \boldsymbol{\varphi A} + \boldsymbol{\psi B} \geq \boldsymbol{\theta} \tag{36d}$$

On the right we wrote the LP dual, such that in (36b-d) a constraint and its Lagrange multiplier are always on a same line ('$\lesseqgtr \boldsymbol{0}$' means that the variable vector is unconstrained). By eliminating the variables $\boldsymbol{\psi}$, the dual reads

$$\min_{\boldsymbol{\varphi}} \sum_{A \subseteq V} \max_{x_A} \theta_A^{\boldsymbol{\varphi}}(x_A) \tag{37}$$

where we abbreviated $\boldsymbol{\theta}^{\boldsymbol{\varphi}} = \boldsymbol{\theta} - \boldsymbol{\varphi}\boldsymbol{A}$. The components of vector $\boldsymbol{\theta}^{\boldsymbol{\varphi}}$ read

$$\theta_A^{\boldsymbol{\varphi}}(x_A) = \theta_A(x_A) - \sum_{B|(B,A)\in J} \varphi_{BA}(x_A) + \sum_{B|(A,B)\in J} \varphi_{AB}(x_B) \tag{38}$$

where $\boldsymbol{\varphi} = \{\, \varphi_{AB}(x_B) \mid (A,B) \in J,\ x_B \in X_B \,\}$. Next we explain the meaning of (38) and (37).

reparameterizations

A *reparameterization* is a transformation of $\boldsymbol{\theta}$ that preserves the objective function $\sum_{A\subseteq V} \theta_A$ of problem (31). The simplest reparameterization is done as follows: pick two interactions θ_A and θ_B with $B \subseteq A$, add an arbitrary function (a 'message') $\varphi_{AB}\colon X_B \to \mathbb{R}$ to θ_A, and subtract it from θ_B:

$$\theta_A(x_A) \leftarrow \theta_A(x_A) + \varphi_{AB}(x_B)\,, \qquad x_A \in X_A\,, \tag{39a}$$

$$\theta_B(x_B) \leftarrow \theta_B(x_B) - \varphi_{AB}(x_B)\,, \qquad x_B \in X_B\,. \tag{39b}$$

E.g., if $A = (1,2,3,4)$ and $B = (2,4)$ then we add a function $\varphi_{1234,24}(x_2,x_4)$ to $\theta_{1234}(x_1,x_2,x_3,x_4)$ and subtract it from $\theta_{24}(x_2,x_4)$. This preserves $\theta_A + \theta_B$ (because φ_{AB} cancels out) and hence also $\sum_{A\subseteq V}\theta_A$. Applying reparameterization (39) to all pairs $(A,B) \in J$ yields (38).

Thus, (38) describes reparameterizations, i.e., for every x_V and $\boldsymbol{\varphi}$ we have

$$\sum_{A\subseteq V} \theta_A(x_A) = \sum_{A\subseteq V} \theta_A^{\boldsymbol{\varphi}}(x_A)\,.$$

Besides this, (38) preserves (for feasible $\boldsymbol{\mu}$) also the objective of the primal program (36): $\boldsymbol{A}\boldsymbol{\mu} = \boldsymbol{0}$ implies $\langle \boldsymbol{\theta}^{\boldsymbol{\varphi}}, \boldsymbol{\mu}\rangle = \langle \boldsymbol{\theta} - \boldsymbol{\varphi}\boldsymbol{A}, \boldsymbol{\mu}\rangle = \langle \boldsymbol{\theta}, \boldsymbol{\mu}\rangle$.

upper bound

By the well-known max-sum dominance, for any $\boldsymbol{\theta}$ we have

$$\max_{x_V} \sum_{A\subseteq V} \theta_A(x_A) \le \sum_{A\subseteq V} \max_{x_A} \theta_A(x_A)\,, \tag{40}$$

thus the right-hand side of (40) is an upper bound on (31). This shows that the *dual (37) minimizes an upper bound on (31) by reparameterizations*.

Note that for each $(A,B) \in J$, marginalization constraint (32a) corresponds via duality to message φ_{AB}. The larger is J, the larger is the set of reparameterizations (38) and hence the smaller the optimal value of (37).

When is inequality (40) (and hence the upper bound) tight? It happens if and only if the independent maximizers of the interactions agree on a common global assignment, i.e., if there exists $y_V \in X_V$ such that

$$y_A \in \operatorname*{argmax}_{x_A} \theta_A(x_A)\,, \qquad A \subseteq V\,.$$

We will further refer to the set $\operatorname{argmax}_{x_A} \theta_A(x_A)$ as the *active joint states* of interaction θ_A. The test can be cast as the *constraint satisfaction problem*

CSP on active joint states

(CSP) (Mackworth, 1991; Rossi et al., 2006) formed by the active joint states of all the interactions (Shlezinger, 1976; Werner, 2007, 2010). Thus, if after solving (37) this CSP is satisfiable for $\boldsymbol{\theta}^{\boldsymbol{\varphi}}$, the relaxation is tight and we have solved our instance of problem (31) exactly. Otherwise, we have only an upper bound on (31).

1.4.4 Max-sum diffusion

fixed point

Max-sum diffusion is a simple convergent 'message-passing' algorithm to tackle the dual LP. It seeks to reparameterize $\boldsymbol{\theta}$ such that

$$\max_{x_{A\setminus B}} \theta_A(x_A) = \theta_B(x_B)\;, \qquad (A,B) \in J,\; x_B \in X_B\;. \tag{41}$$

update

The algorithm repeats the following iteration:

Enforce (41) for a single pair $(A,B) \in J$ by reparameterization (39).

This is done by setting $\varphi_{AB}(x_B) = [\theta_B(x_B) - \max_{x_{A\setminus B}} \theta_A(x_A)]/2$ in (39). The algorithm converges to a fixed point when (41) holds for all $(A,B) \in J$.

We remark that originally (Kovalevsky and Koval, approx. 1975), max-sum diffusion was formulated for problems with only binary (no unary) interactions. The generalization (41) by Werner (2008a, 2010) is interesting in the fact that (41) has exactly the same form as (32a). We pursued this idea further in (Werner, 2008b).

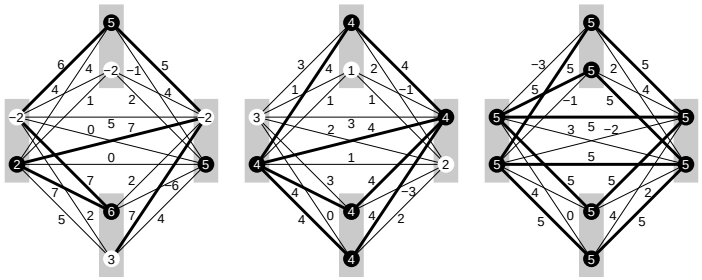
Doing reparameterization by messages rather than by modifying $\boldsymbol{\theta}$ yields Algorithm 1.6. To correctly handle infinite weights, the algorithm expects that $[\theta_B(x_B) > -\infty] \Leftrightarrow [\max_{x_{A\setminus B}} \theta_A(x_A) > -\infty]$ for every $(A,B) \in J$.

Algorithm 1.6 Max-sum diffusion

repeat
  for $(A,B) \in J$ and $x_B \in X_B$ such that $\theta_B(x_B) > -\infty$ **do**
    $\varphi_{AB}(x_B) \leftarrow \varphi_{AB}(x_B) + [\theta^{\boldsymbol{\varphi}}_B(x_B) - \max_{x_{A\setminus B}} \theta^{\boldsymbol{\varphi}}_A(x_A)]/2$
  end for
until convergence

properties

The diffusion iteration decreases or preserves, but never increases, the upper bound. In general, the algorithm does not find the global minimum of (37) but only a certain local minimum (where 'local' is meant w.r.t. block-coordinate moves), which is nevertheless very good in practice. These local minima are characterized by *local consistency* (Rossi et al., 2006, chapter 3) of the CSP formed by the active joint states.



(a) (b) (c)

Figure 1.3: The visualization of a problem with $|X_v| = 2$ variable states and hypergraph E as in Figure 1.2. The variables are shown as boxes, their numbering is $\begin{smallmatrix} & 1 & \\ 2 & & 4 \\ & 3 & \end{smallmatrix}$. Variable states are shown as circles, joint states of variable pairs as edges. Weights $\theta_A(x_A)$, $A \in E$, are written in the circles and next to the edges. Active joint states are emphasized (black circles, thick edges). Example (a) is not a diffusion fixed point, (b,c) are diffusion fixed points for J from Figure 1.2a. Examples (a,b) are reparameterizations of each other (this is not obvious at the first sight), (c) is not a reparameterization of (a,b). For (b), a global assignment x_V can be composed of the active joint states and hence inequality (40) is tight. For (a,c), no global assignment x_V can be composed of the active joint states, hence inequality (40) is not tight.

Note that the only non-trivial operation in Algorithm 1.6 is computing the max-marginals $\max_{x_{A \setminus B}} \theta_A^{\boldsymbol{\varphi}}(x_A)$. By (38), this is an instance of problem (31). When $|A|$ is small (such as for a binary interaction), computing the max-marginals is trivial. But even when $|A|$ is large, depending on the function θ_A and on J there may exist an algorithm polynomial in $|A|$ to compute $\max_{x_{A \setminus B}} \theta_A^{\boldsymbol{\varphi}}(x_A)$. In that case, Algorithm 1.6 still can be used.

If $\theta_A = 0$, it depends only on J whether $\max_{x_{A \setminus B}} \theta_A^{\boldsymbol{\varphi}}(x_A)$ can be computed in polynomial time. E.g., in Figure 1.2c we have $\theta_{1234} = 0$ and hence, by (38), $\theta_{1234}^{\boldsymbol{\varphi}}(x_1, x_2, x_3, x_4) = \varphi_{1234,12}(x_1, x_2) + \varphi_{1234,23}(x_2, x_3) + \varphi_{1234,34}(x_3, x_4) + \varphi_{1234,14}(x_1, x_4)$. Thus we have a problem on a cycle, which can be solved more efficiently than by going through all states (x_1, x_2, x_3, x_4).

diffusion solves subproblems

This suggests that diffusion in a sense exactly solves certain small subproblems (which links it to the dual decomposition interpretation (Komodakis et al., 2007)). This can be formalized as follows. Let $A \in F \subseteq 2^A$. Clearly,

$$\max_{x_A} \sum_{B \in F} \theta_B(x_B) \leq \sum_{B \in F} \max_{x_B} \theta_B(x_B) \tag{42}$$

for any $\boldsymbol{\theta}$, which is inequality (40) written for subproblem F. Let $J = \{\, (A, B) \mid B \in F \,\}$. In this case, the minimal upper bound for subproblem F is tight. To see it, just do reparameterization (39) with $\varphi_{AB} = \theta_B$ for

$B \in F$, which results in $\theta_B = 0$ for $B \in F \setminus \{A\}$, hence (42) is tight trivially. What is not self-evident is that diffusion finds the *global* minimum in this case. It does: if $\boldsymbol{\theta}$ satisfies (41) for $J = \{\, (A, B) \mid B \in F \,\}$ then (42) is tight.

1.4.5 Dual cutting plane algorithm

incremental scheme

The relaxation can be tightened *incrementally* during dual optimization. Any time during Algorithm 1.6, the current J can be extended by any $J' \subseteq I$, $J' \cap J = \emptyset$. The messages φ_{AB} for $(A, B) \in J'$ are initialized to zero. Clearly, this does not change the current upper bound. Future diffusion iterations can only preserve or improve the bound, so the scheme remains monotonic. This can be imagined as if the added variables φ_{AB} extended the space of possible reparameterizations and diffusion is now trying to take advantage of it. If the bound does not improve, all we will have lost is the memory occupied by the added variables. Algorithm 1.7 describes this.

In the primal domain, this incremental scheme can be understood as a cutting plane algorithm. We discuss this later in Section 1.4.6.

Algorithm 1.7 Dual cutting plane algorithm

1: **Initialization:** Choose $J \subseteq I$ and $\mathcal{J} \subseteq 2^I$.
2: **repeat**
3: Execute any number of iterations of Algorithm 1.6.
4: *Separation oracle:* Choose $J' \in \mathcal{J}$, $J \cap J' = \emptyset$.
5: $J \leftarrow J \cup J'$
6: Allocate messages φ_{AB}, $(A, B) \in J'$, and set them to zero.
7: **until** no suitable J' can be found

separation test

On line 4 of Algorithm 1.7 the separation oracle is called, which chooses a promising extension J' from some predefined set $\mathcal{J} \subseteq 2^I$ of candidate extensions. We assume $|\mathcal{J}|$ is small so that $\mathcal{J}$ it can be searched exhaustively. For that, we need a test to recognize whether a given J' would lead to a (good) bound improvement. We refer to this as the *separation test*.

Of course, a trivial necessary and sufficient separation test is to extend J by J' and run diffusion till convergence. One easily invents a faster test:

> *Execute several diffusion iterations only on pairs J'. If this improves the bound, then running diffusion on $J \cup J'$ would inevitably improve the bound too.*

This local test is sufficient but not necessary for improvement because even if running diffusion on J' does not improve the bound, it may change the problem such that future diffusion iterations on $J \cup J'$ improve it.

greediness

Even with a sufficient and necessary separation test, Algorithm 1.7 is

'greedy' in the following sense. For $J'_1, J'_2 \subseteq I$, it can happen that extending J by J'_1 alone or by J'_2 alone does not lead to a bound improvement but extending J by $J'_1 \cup J'_2$ does. See (Werner, 2010) for an example.

The extension J' can be an arbitrary subset of I. One form of extension has a clear meaning: pick a hyperedge A not yet coupled to any other hyperedge, choose $F \subseteq 2^A$, and let $J' = \{\,(A, B) \mid B \in F\,\}$. This can be seen as 'connecting' a so far disconnected interaction θ_A to the problem.

adding zero subproblems

An important special case is connecting a zero interaction, $\theta_A = 0$. Because, by (38), we have $\theta^{\boldsymbol{\varphi}}_A(x_A) = \sum_{B \in F} \varphi_{AB}(x_B)$, we refer to this extension as *adding a zero subproblem* F. In this case, the separation test can be done more efficiently than by running diffusion on J'. This is based on the fact stated at the end of Section 1.4.4: if inequality (42) is not tight for current $\boldsymbol{\theta}^{\boldsymbol{\varphi}}$ then running diffusion on J' will surely make it tight, i.e., improve the bound. Note, we do not need $A \in F$ here because $\theta_A = 0$. The gap in (42) is an estimate of the expected improvement.

This has a clear interpretation in CSP terms. Inequality (42) is tight if and only if the CSP formed by the active joint states of interactions F is satisfiable. If this CSP is unsatisfiable then J' will improve the bound. Therefore, *the separation oracle needs to find a (small) unsatisfiable subproblem of the CSP formed by the active joint states.*

For instance, Figure 1.3c shows a problem after diffusion convergence, for J defined by Figure 1.2a. The CSP formed by the active joint states is not satisfiable because it contains an unsatisfiable subproblem, the cycle $F = \{(1,2),(1,4),(2,4)\}$. Hence, adding zero subproblem F (which yields J from Figure 1.2b) and running diffusion would improve the bound. Adding the zero cycle $F = \{(1,2),(1,4),(2,3),(3,4)\}$ (yielding J from Figure 1.2c) or the whole zero problem $F = E$ would improve the bound too.

Figure 1.4 shows a more complex example.

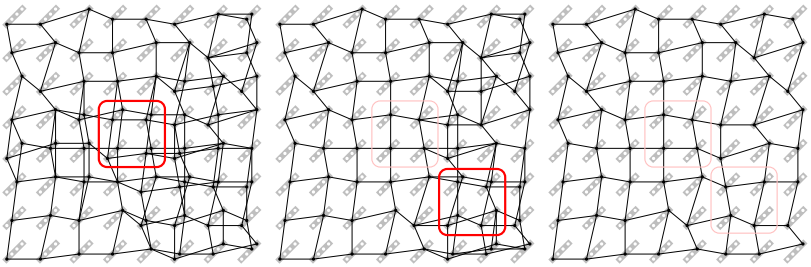
removing zero subproblems

Message passing algorithms have a drawback: after extending J, they need a long time to re-converge. This can be partially alleviated by adding multiple subproblems at a time and doing so before full convergence. As some of the added subproblems might later turn out redundant, we found helpful to remove redundant subproblems occasionally – which can be done without sacrificing monotonicity of bound improvement. This is a (dual) way of 'constraint management', often used in cutting plane methods.

1.4.6 Zero interactions as projection, marginal polytope

zero interactions act as projection

In the beginning, formula (31), we added all possible zero interactions to our problem. This has proved natural because the problem is after all defined only up to reparameterizations and thus any zero interaction can become



(a) (b) (c)

Figure 1.4: Two steps of the cutting plane algorithm for a problem with the 8×8 grid graph E and $|X_v| = 4$ variable states. The set $\mathcal{J}$ of candidate extensions contains all cycles of length 4. Only the active joint states are shown. Subfigure (a) shows the problem after diffusion has converged for $J = \{\, (A, B) \mid B \subseteq A;\ A, B \in E \,\}$. The upper bound is not tight because of the depicted unsatisfiable subproblem (an inconsistent cycle). Adding the cycle and letting diffusion re-converge results in problem (b) with a better bound. The original cycle is now satisfiable but a new unsatisfiable cycle occurred. Adding this cycle solves the problem, (c).

non-zero. Now, let us see how the LP relaxation would look like without adopting this abstraction. Let $T(E) = \{\, (A, x_A) \mid A \in E,\ x_A \in X_A \,\}$ denote the restriction of the set T to hypergraph E. Since zero interactions do not contribute to the objective function of (35), (35) can be written as

$$\max\{\, \langle \boldsymbol{\theta}, \boldsymbol{\mu} \rangle \mid \boldsymbol{\mu} \in \mathcal{M}(J) \,\} = \max\{\, \langle \pi_{T(E)} \boldsymbol{\theta},\, \boldsymbol{\mu} \rangle \mid \boldsymbol{\mu} \in \pi_{T(E)} \mathcal{M}(J) \,\} \quad (43)$$

where $\pi_{D'} \boldsymbol{a} \in \overline{\mathbb{R}}^{D'}$ denotes the projection of a vector $\boldsymbol{a} \in \overline{\mathbb{R}}^{D}$ on dimensions $D' \subseteq D$, thus $\pi_{D'}$ deletes the components $D \setminus D'$ of $\boldsymbol{a}$. Applied to a set of vectors, $\pi_{D'}$ does this for every vector in the set. Informally, (43) shows that *zero interactions act as the projection of the feasible set onto the space of non-zero interactions*.

marginal polytope

The set $\pi_{T(E)} \mathcal{M} \subseteq [0, 1]^{T(E)}$ is recognized as the *marginal polytope* (Wainwright et al., 2005) of hypergraph E. Its elements $\boldsymbol{\mu}$ are the marginals over variable subsets E of some global distribution μ_V, which not necessarily is part of $\boldsymbol{\mu}$. The marginal polytope of the complete hypergraph $\pi_{T(2^V)} \mathcal{M} = \mathcal{M}$ is of fundamental importance because all other marginal polytopes are its projections. For $J \subseteq I$, the set $\pi_{T(E)} \mathcal{M}(J) \supseteq \pi_{T(E)} \mathcal{M}$ is a relaxation of the marginal polytope, which may contain elements $\boldsymbol{\mu}$ that no longer can be realized as the marginals of any global distribution μ_V.

While $\operatorname{conv} \boldsymbol{\delta}(X_V) = \mathcal{M}$ is the integral hull of the problem $\max\{\, \langle \boldsymbol{\theta}, \boldsymbol{\mu} \rangle \mid \boldsymbol{\mu} \in \boldsymbol{\delta}(X_V) \,\}$, the polytope $\operatorname{conv} \pi_{T(E)} \boldsymbol{\delta}(X_V) = \pi_{T(E)} \operatorname{conv} \boldsymbol{\delta}(X_V) = \pi_{T(E)} \mathcal{M}$ is the integral hull of the problem $\max\{\, \langle \pi_{T(E)} \boldsymbol{\theta}, \boldsymbol{\mu} \rangle \mid \boldsymbol{\mu} \in \pi_{T(E)} \boldsymbol{\delta}(X_V) \,\}$.

local vs. non-local relaxations

Following (Wainwright et al., 2005), we say a relaxation J is *local in* E if $A, B \in E$ for every $(A, B) \in J$. E.g., in Figure 1.2 only relaxation (a) is local. For local relaxations, the distributions μ_A, $A \notin E$, are not coupled to any other distributions and the action of $\pi_{T(E)}$ on $\mathcal{M}(J)$ is simple: it just removes these superfluous coordinates. Thus, $\pi_{T(E)}\mathcal{M}(J)$ has an explicit description by a small (polynomial in $|E|$) number of linear constraints.

For non-local relaxations, the effect of the projection is in general complex and the number of facets of $\pi_{T(E)}\mathcal{M}(J)$ is exponential in $|E|$. It is well-known that to compute the explicit description of a projection of a polyhedron can be extremely difficult – which suggests that to directly look for the facets of $\pi_{T(E)}\mathcal{M}$ might be a bad idea. Non-local relaxations can be seen as a *lift-and-project approach*: we lift from dimensions $T(E)$ to dimensions T, impose constraints in this lifted space, and project back onto dimensions $T(E)$.

primal view on the cutting plane algorithm

Now it is clear what is the geometry of our cutting plane algorithm in the primal space $[0, 1]^{T(E)}$. Suppose max-sum diffusion has found a global optimum of the dual and let $\boldsymbol{\mu}^* \in [0, 1]^{T(E)}$ be a corresponding primal optimum. A successful extension of J means that a set (perhaps exponentially large) of cutting planes is added to the primal that separates $\boldsymbol{\mu}^*$ from $\pi_{T(E)}\mathcal{M}$. However, $\boldsymbol{\mu}^*$ is not computed explicitly at all (and, let us remark, it is expensive to compute $\boldsymbol{\mu}^*$ from a dual optimum for large problems). In fact, $\boldsymbol{\mu}^*$ may not even exist because diffusion may find only a local optimum of the dual – we even need not run diffusion to full convergence.

1.4.7 Conclusions

We have presented the theory of the cutting plane approach to the MAP inference problem, as well as a very general message passing algorithm to implement this approach. In comparison with other similar works, the theory, and Algorithm 1.6 in particular, is very simple. We have shown that for the case of adding subproblems, separation means finding a (small) unsatisfiable subproblem of the CSP formed by the active joint states.

We assumed, in Section 1.4.5, that the set $\mathcal{J}$ of candidate extensions is tractably small. Is there a polynomial algorithm to select an extension from an intractably large set $\mathcal{J}$? In particular, is there a polynomial algorithm to find a small unsatisfiable subproblem (most interestingly, a cycle) in a given CSP? This is currently an open problem. An inspiration for finding such algorithms are local consistencies in CSP (Rossi et al., 2006, chapter 3).

Let us remark that several polynomial algorithms are known to separate intractable families of cutting planes of the *max-cut polytope* (Deza and Laurent, 1997), closely related to the marginal polytope. Some of them have been applied to MAP inference by Sontag and Jaakkola (2007) and Sontag

(2007). As these algorithms work in the primal space, they cannot be used in our dual cutting plane scheme – we need a *dual separation algorithm*.

References

A. Argyriou, R. Hauser, C. A. Micchelli, and M. Pontil. A dc-programming algorithm for kernel selection. In *Proc. Intl. Conf. Machine Learning*, pages 41–48. ACM Press, 2006.

F. Bach. Exploring large feature spaces with hierarchical multiple kernel learning. In M. C. Mozer, M. I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems 9*, pages 105–112, 2009.

F. R. Bach, G. R. G. Lanckriet, and M. I. Jordan. Multiple kernel learning, conic duality, and the smo algorithm. In *Proc. Intl. Conf. Machine Learning*. ACM Press, 2004.

D. P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, MA, 1999. ISBN 1-886529-00-0.

S. Boyd and L. Vandenberge. Localization and cutting-plane methods. Stanford University, California, USA, Unpublished lecture notes, 2008. URL `http://see.stanford.edu/materials/lsocoee364b/05-localization_methods_notes.pdf`.

S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, March 2004. ISBN 0521833787.

E. W. Cheney and A. A. Goldstein. Newton's method for convex programming and Tchebycheff approximation. *Numerische Mathematik*, 1:253–268, 1959. ISSN 0029-599X.

C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20 (3):273–297, 1995.

C. Cortes, M. Mohri, and A. Rostamizadeh. Learning non-linear combinations of kernels. In M. C. Mozer, M. I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems 9*, pages 396–404, 2009.

G. Dantzig, R. Fulkerson, and S. Johnson. Solution of a large-scale traveling-salesman problem. *Operations Research*, 2:393–410, 1954.

M. M. Deza and M. Laurent. *Geometry of Cuts and Metrics*. Springer, Berlin, 1997.

V. Franc and S. Sonnenburg. OCAS optimized cutting plane algorithm for support vector machines. In *Proc. Intl. Conf. Machine Learning*, pages

320–327. ACM Press, 2008.

V. Franc and S. Sonnenburg. Optimized cutting plane algorithm for large-scale risk minimization. *Journal of Machine Learning Research*, 10:2157–2192, 2010.

M. Grötschel, L. Lovász, and A. Schrijver. The ellipsoid method and its consequences in combinatorial optimization. *Combinatorica*, 1(2):169–197, 1981.

T. Joachims. Making large–scale SVM learning practical. In B. Schölkopf, C. Burges, and A. Smola, editors, *Advances in Kernel Methods — Support Vector Learning*, pages 169–184, Cambridge, MA, USA, 1999. MIT Press.

J. K. Johnson, D. M. Malioutov, and A. S. Willsky. Lagrangian relaxation for MAP estimation in graphical models. In *Allerton Conf. Communication, Control and Computing*, 2007.

J. E. Kelley. The cutting-plane method for solving convex programs. *Journal of the Society for Industrial and Applied Mathematics*, 8(4):703–712, 1960.

K. C. Kiwiel. An aggregate subgradient method for nonsmooth convex minimization. *Mathematical Programming*, 27:320–341, 1983.

M. Kloft, U. Brefeld, S. Sonnenburg, P. Laskov, K.-R. Müller, and A. Zien. Efficient and accurate lp-norm multiple kernel learning. In M. C. Mozer, M. I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems 9*, pages 997–1005. MIT Press, 2009.

N. Komodakis and N. Paragios. Beyond loose LP-relaxations: Optimizing MRFs by repairing cycles. In *European Conf. on Computer Vision*, 2008.

N. Komodakis, N. Paragios, and G. Tziritas. MRF optimization via dual decomposition: Message-passing revisited. In *Proc. Intl. Conf. Computer Vision*, 2007.

A. Koster, C. P. M. van Hoesel, and A. W. J. Kolen. The partial constraint satisfaction problem: Facets and lifting theorems. *Operations Research Letters*, 23(3–5):89–97, 1998.

V. A. Kovalevsky and V. K. Koval. A diffusion algorithm for decreasing the energy of the max-sum labeling problem. Glushkov Institute of Cybernetics, Kiev, USSR. Unpublished, personally communicated to T. Werner by M. I. Schlesinger., approx. 1975.

M. P. Kumar and P. H. S. Torr. Efficiently solving convex relaxations for MAP estimation. In *Proc. Intl. Conf. Machine Learning*, pages 680–687. ACM, 2008.

G. Lanckriet, T. D. Bie, N. Cristianini, M. Jordan, and W. Noble. A statistical framework for genomic data fusion. *Bioinformatics*, 20:2626–

2635, 2004a.

G. Lanckriet, N. Cristianini, L. E. Ghaoui, P. Bartlett, and M. I. Jordan. Learning the kernel matrix with semi-definite programming. *Journal of Machine Learning Research*, 5:27–72, 2004b.

C. Lemaréchal, A. Nemirovskii, and Y. Nesterov. New variants of bundle methods. *Mathematical Programming*, 69(1–3):111–147, 1995.

A. Mackworth. Constraint satisfaction. In *Encyclopaedia of Artificial Intelligence*, pages 285–292. John Wiley, 1991.

J. S. Nath, G. Dinesh, S. Ramanand, C. Bhattacharyya, A. Ben-Tal, and K. R. Ramakrishnan. On the algorithmics and applications of a mixed-norm based kernel learning formulation. In M. C. Mozer, M. I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems 9*, pages 844–852, 2009.

A. S. Nemirovskij and D. B. Yudin. *Problem Complexity and Method Efficiency in Optimization*. Wiley Interscience, New York, 1983.

A. Rakotomamonjy, F. Bach, S. Canu, and Y. Grandvalet. More efficiency in multiple kernel learning. In *Proc. Intl. Conf. Machine Learning*, pages 775–782, 2007.

A. Rakotomamonjy, F. R. Bach, S. Canu, and Y. Grandvalet. SimpleMKL. *Journal of Machine Learning Research*, 9:2491–2521, 2008.

F. Rossi, P. van Beek, and T. Walsh. *Handbook of Constraint Programming*. Elsevier, 2006.

B. Schölkopf and A. Smola. *Learning with Kernels*. MIT Press, 2002.

A. Schrijver. *Combinatorial Optimization : Polyhedra and Efficiency (Algorithms and Combinatorics)*. Springer, 2003.

M. I. Shlezinger. Syntactic analysis of two-dimensional visual signals in noisy conditions. *Cybernetics and Systems Analysis*, 12(4):612–628, 1976. Translation from Russian.

S. Sonnenburg, G. Rätsch, and C. Schäfer. Learning interpretable SVMs for biological sequence classification. In S. Miyano, J. P. Mesirov, S. Kasif, S. Istrail, P. A. Pevzner, and M. Waterman, editors, *Research in Computational Molecular Biology, 9th Annual International Conference (RECOMB)*, volume 3500, pages 389–407. Springer-Verlag, 2005.

S. Sonnenburg, G. Rätsch, C. Schäfer, and B. Schölkopf. Large Scale Multiple Kernel Learning. *Journal of Machine Learning Research*, 7:1531–1565, July 2006a.

S. Sonnenburg, A. Zien, and G. Rätsch. ARTS: Accurate Recognition of Transcription Starts in Human. *Bioinformatics*, 22(14):e472–e480, 2006b.

S. Sonnenburg, G. Rätsch, S. Henschel, C. Widmer, J. Behr, A. Zien, F. de Bona, A. Binder, C. Gehl, and V. Franc. The SHOGUN machine learning toolbox. *Journal of Machine Learning Research*, 11:1799–1802, June 2010. URL `http://www.shogun-toolbox.org`.

D. Sontag. Cutting plane algorithms for variational inference in graphical models. Master's thesis, Massachusetts Institute of Technology, Department of Electrical Engineering and Computer Science, 2007.

D. Sontag and T. Jaakkola. New outer bounds on the marginal polytope. In *Advances in Neural Information Processing Systems 7*, 2007.

D. Sontag, T. Meltzer, A. Globerson, T. Jaakkola, and Y. Weiss. Tightening LP relaxations for MAP using message passing. In *Conf. Uncertainty in Artificial Intelligence (UAI)*, 2008.

N. Subrahmanya and Y. C. Shin. Sparse multiple kernel learning for signal processing applications. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32:788–798, 2010.

M. Szafranski, Y. Grandvalet, and A. Rakotomamonjy. Composite kernel learning. In *Proc. Intl. Conf. Machine Learning*, 2008.

C. Teo, Q. Le, A. Smola, and S. Vishwanathan. A scalable modular convex solver for regularized risk minimization. In *Proc. Intl. Conf. Knowledge Discovery and Data Mining*, 2007.

C. Teo, S. Vishwanathan, A. Smola, and V. Quoc. Bundle methods for regularized risk minimization. *Journal of Machine Learning Research*, 11: 311–365, 2010.

I. Tsochantaridis, T. Joachims, T. Hofmann, and Y. Altun. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484, 2005.

M. Varma and B. R. Babu. More generality in efficient multiple kernel learning. In *Proc. Intl. Conf. Machine Learning*, pages 1065–1072, New York, NY, USA, 2009. ACM Press.

M. Wainwright, T. Jaakkola, and A. Willsky. MAP estimation via agreement on (hyper)trees: message passing and linear programming approaches. *IEEE Trans. Information Theory*, 51(11):3697–3717, 2005.

M. J. Wainwright and M. I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305, 2008.

T. Werner. High-arity interactions, polyhedral relaxations, and cutting plane algorithm for soft constraint optimisation (MAP-MRF). In *Proc. IEEE Conf. Computer Vision and Pattern Recognition*, June 2008a.

T. Werner. Revisiting the linear programming relaxation approach to Gibbs energy minimization and weighted constraint satisfaction. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, August 2010.

T. Werner. A linear programming approach to max-sum problem: A review. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(7): 1165–1179, July 2007.

T. Werner. Marginal consistency: Unifying constraint propagation on commutative semirings. In *Intl. Workshop on Preferences and Soft Constraints (co-located with Conf. on Principles and Practice of Constraint Programming)*, pages 43–57, September 2008b.

T. Werner. A linear programming approach to max-sum problem: A review. Technical Report CTU–CMP–2005–25, Center for Machine Perception, Czech Technical University, December 2005.

Z. Xu, R. Jin, I. King, and M. Lyu. An extended level method for efficient multiple kernel learning. In M. C. Mozer, M. I. Jordan, and T. Petsche, editors, *Advances in Neural Information Processing Systems 9*, pages 1825–1832, 2009a.

Z. Xu, R. Jin, J. Ye, M. R. Lyu, and I. King. Non-monotonic feature selection. In *Proc. Intl. Conf. Machine Learning*, pages 1145–1152, 2009b.

A. Zien and C. S. Ong. Multiclass multiple kernel learning. In *Proc. Intl. Conf. Machine Learning*, pages 1191–1198. ACM Press, 2007.

Universality of the Local Marginal Polytope

Daniel Průša and Tomáš Werner

Abstract—We show that solving the LP relaxation of the min-sum labeling problem (also known as MAP inference problem in graphical models, discrete energy minimization, or valued constraint satisfaction) is not easier than solving any linear program. Precisely, every polytope is linear-time representable by a local marginal polytope and every LP can be reduced in linear time to a linear optimization (allowing infinite costs) over a local marginal polytope. The reduction can be done (though with a higher time complexity) even if the local marginal polytope is restricted to have a planar structure.

Index Terms—graphical model, Markov random field, discrete energy minimization, valued constraint satisfaction, linear programming relaxation, local marginal polytope

1 Introduction

The *min-sum (labeling) problem* is defined as follows: given a set of discrete variables and a set of functions depending on one or two variables, minimize the sum of the functions over all variables. This problem arises in MAP inference in graphical models [22] and it is also known as discrete energy minimization [9] or valued constraint satisfaction [21].

This NP-complete problem has a natural linear programming (LP) relaxation, proposed by a number of authors [18], [13], [4], [22]. This relaxation is equivalent to the dual (Lagrangian) decomposition of the min-sum problem [8], [12], [19]. While the min-sum problem can be formulated as a linear optimization over the *marginal polytope*, the LP relaxation approximates this polytope by its outer bound, the *local marginal polytope* [22].

The relaxation is exact for a large class of min-sum instances and it is a basis for constructing good approximations for many other instances [20], [23], [9]. It is therefore of great practical interest to have efficient algorithms to solve the LP relaxation.

To solve the LP relaxation, the simplex and interior point methods are prohibitively inefficient for large-scale instances (which often occur, e.g., in computer vision). For min-sum problems with 2 labels, the LP relaxation can be solved efficiently because it reduces in linear time to max-flow [3], [17]. For more general problems, no really efficient algorithm is known to solve the LP.

In this article we show that the quest for efficient algorithms to solve the LP relaxation of the general min-sum problem has a fundamental limitation, because this task is not easier than solving any linear program. Precisely, we prove the following theorems.

Theorem 1. *Every polytope is (up to scale) a coordinate-erasing projection of a face of a local marginal polytope with 3 labels, whose description can be computed from the input polytope in linear time.*

The input polytope is described by a set of linear inequalities with integer coefficients. By coordinate-erasing projection, we mean a projection that copies a subset of coordinates and erases the remaining ones.

Theorem 2. *Every linear program can be reduced in linear time to a linear optimization (allowing infinite costs) over a local marginal polytope with 3 labels.*

While Theorem 2 immediately follows from Theorem 1, the situation is more complex when infinite costs are not allowed. In this case, the reduction time and the output size are quadratic (see Theorem 9).

Given these negative results, one may ask whether the LP relaxation can be solved efficiently for some useful subclasses of the min-sum problem. One such subclass is the planar min-sum problem, which frequently occurs in computer vision. We show (in Theorem 11) that even in this case, the reduction can be done (with infinite costs allowed), in better than quadratic time.

Similar universality results are known also for other polytopes, e.g., the three-way transportation polytope [6] and the traveling salesman polytope [2].

2 The Local Marginal Polytope

Let (V, E) be an undirected graph, where V is a finite set of *objects* and $E \subseteq \binom{V}{2}$ is a set of object pairs. Let K be a finite set of *labels*. Let $g_u\colon K \to \overline{\mathbb{R}}$ and $g_{uv}\colon K \times K \to \overline{\mathbb{R}}$ be unary and binary *cost functions*, where $\overline{\mathbb{R}} = \mathbb{R} \cup \{\infty\}$ and we adopt that $g_{uv}(k, \ell) = g_{vu}(\ell, k)$. The *min-sum problem* is defined as

$$\min_{\mathbf{k} \in K^V} \Big(\sum_{u \in V} g_u(k_u) + \sum_{\{u,v\} \in E} g_{uv}(k_u, k_v) \Big). \tag{1}$$

All the costs $g_u(k), g_{uv}(k, \ell)$ form a vector $\mathbf{g} \in \overline{\mathbb{R}}^I$ where $I = (V \times K) \cup \{\, \{(u, k), (v, \ell)\} \mid \{u, v\} \in E;\ k, \ell \in K \,\}$. The problem instance is given by a tuple $(V, E, K, \mathbf{g})$.

• *The authors are with the Department of Cybernetics, Czech Technical University, Karlovo náměstí 13, 12135 Praha, Czech Republic. Emails: {prusapa1,werner}@cmp.felk.cvut.cz.*

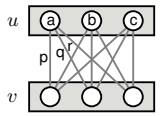


Fig. 1. A pair of objects $\{u, v\} \in E$ with $|K| = 3$ labels.

The *local marginal polytope* [22] is the set Λ of vectors $\boldsymbol{\mu} \in \mathbb{R}_+^I$ satisfying

$$\sum_{\ell \in K} \mu_{uv}(k, \ell) = \mu_u(k), \quad u \in V,\ v \in N_u,\ k \in K \tag{2a}$$

$$\sum_{k \in K} \mu_u(k) = 1, \qquad u \in V \tag{2b}$$

where $N_u = \{\, v \mid \{u, v\} \in E \,\}$ are the neighbors of u and we assume $\mu_{uv}(k, \ell) = \mu_{vu}(\ell, k)$. The numbers $\mu_u(k), \mu_{uv}(k, \ell)$ are known as *pseudomarginals* [22]. The local marginal polytope is given by a triplet (V, E, K).

The LP relaxation of the min-sum problem reads

$$\Lambda^*(\mathbf{g}) = \operatorname*{argmin}_{\boldsymbol{\mu} \in \Lambda} \langle \mathbf{g}, \boldsymbol{\mu} \rangle \tag{3}$$

where in the scalar product $\langle \mathbf{g}, \boldsymbol{\mu} \rangle$ we define $0 \cdot \infty = 0$. The set (3) contains all vectors $\boldsymbol{\mu}$ for which $\langle \mathbf{g}, \boldsymbol{\mu} \rangle$ attains minimum over Λ. It is itself a polytope, a face of Λ.

We will depict min-sum problems by diagrams, as in Figure 1. Objects $u \in V$ are depicted as boxes, labels $(u, k) \in I$ as nodes, label pairs $\{(u, k), (v, \ell)\} \in I$ as edges. Each node is assigned a unary pseudomarginal $\mu_u(k)$ and cost $g_u(k)$. Each edge is assigned a binary pseudomarginal $\mu_{uv}(k, \ell)$ and cost $g_{uv}(k, \ell)$.

Note the meaning of constraints (2) in Figure 1. Constraint (2b) imposes for unary pseudomarginals a, b, c that $a + b + c = 1$. Constraint (2a) imposes for binary pseudomarginals p, q, r that $a = p + q + r$.

3 Input Polyhedron

We consider the input polyhedron in the form

$$P = \{\, \mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n \mid \mathbf{A}\mathbf{x} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0} \,\} \tag{4}$$

where[1] $\mathbf{A} = [a_{ij}] \in \mathbb{Z}^{m \times n}$, $\mathbf{b} = (b_1, \ldots, b_m) \in \mathbb{Z}^m$, $m \leq n$. We assume there is at least one non-zero entry in each row and column of $\mathbf{A}$. The instance of polyhedron (4) is given by $(\mathbf{A}, \mathbf{b})$ or, in short, by the extended matrix

$$\bar{\mathbf{A}} = [\bar{a}_{ij}] = [\mathbf{A} \,|\, \mathbf{b}] \in \mathbb{Z}^{m \times (n+1)}. \tag{5}$$

It will be convenient to rewrite the system $\mathbf{A}\mathbf{x} = \mathbf{b}$ as follows. In the i-th equation

$$a_{i1}x_1 + \cdots + a_{in}x_n = b_i \tag{6}$$

it is assumed that $b_i \geq 0$ (if not, multiply the equation by -1). Further, the terms with negative coefficients are moved to the right-hand side, such that both sides have only non-negative terms. Thus, (6) is rewritten as

$$a_{i1}^+ x_1 + \cdots + a_{in}^+ x_n = a_{i1}^- x_1 + \cdots + a_{in}^- x_n + b_i \tag{7}$$

where $a_{ij}^+ \geq 0$, $a_{ij}^- \geq 0$, $a_{ij} = a_{ij}^+ - a_{ij}^-$. We assume w.l.o.g. that $a_{i1}^+ + \cdots + a_{in}^+ \neq 0$ and $a_{i1}^- + \cdots + a_{in}^- + b_i \neq 0$.

The following lemmas give some bounds that will be needed in the encoding algorithm.

Lemma 3. *For every matrix $\mathbf{A} \in \mathbb{R}^{n \times n}$ with columns $\mathbf{a}_j$,*

$$|\det \mathbf{A}| \leq \prod_{j=1}^{n} \|\mathbf{a}_j\|_2 \leq \prod_{j=1}^{n} \|\mathbf{a}_j\|_1.$$

Proof: The first inequality is well-known as Hadamard's inequality. The second inequality holds because $\|\mathbf{a}\|_2 \leq \|\mathbf{a}\|_1$ for every $\mathbf{a} \in \mathbb{R}^n$. □

Lemma 4. *Let $\mathbf{b} \neq \mathbf{0}$. Let $(x_1, \ldots, x_n)$ be a vertex of P. Then for each j we have $x_j = 0$ or $M^{-1} \leq x_j \leq M$ where*

$$M = \prod_{j=1}^{n+1} \sum_{i=1}^{m} |\bar{a}_{ij}|. \tag{8}$$

Proof: It is well-known from the theory of linear programming that every vertex $\mathbf{x}$ of P is a solution of a system $\mathbf{A}'\mathbf{x}' = \mathbf{b}'$, where $\mathbf{x}' = (x'_1, x'_2, \ldots)$ are the non-zero components of $\mathbf{x}$, $\mathbf{A}'$ is a non-singular submatrix of $\mathbf{A}$, and $\mathbf{b}'$ is a subvector of $\mathbf{b}$. By Cramer's rule,

$$x'_j = \frac{\det \mathbf{A}'_j}{\det \mathbf{A}'} \tag{9}$$

where $\mathbf{A}'_j$ denotes $\mathbf{A}'$ with the j-th column replaced by $\mathbf{b}'$. Lemma 3 implies $|\det \mathbf{A}'_j|, |\det \mathbf{A}'| \leq M$. □

Lemma 5. *Let P be bounded. Then for every $\mathbf{x} \in P$, each side of equation (7) is not greater than*

$$N = M \max_{i=1}^{m} \sum_{j=1}^{n} |a_{ij}|. \tag{10}$$

Proof: Since every point $(x_1, \ldots, x_n)$ of P is a convex combination of vertices of P, we have $x_j \leq M$ for each j. Hence, $a_{i1}^+ x_1 + \cdots + a_{in}^+ x_n \leq M(|a_{i1}| + \cdots + |a_{in}|) \leq N$ for each i. □

4 Encoding a Polytope

In this section, we prove Theorem 1 by constructing, in linear time, a min-sum problem $(V, E, K, \mathbf{g})$ with costs $\mathbf{g} \in \{0, 1\}^I$ such that the input polyhedron P is a scaled coordinate-erasing projection of $\Lambda^*(\mathbf{g})$. We assume that P is bounded, i.e., a polytope[2].

1. The assumption that $(\mathbf{A}, \mathbf{b})$ are integer-valued is common, see e.g. [10]. In the more general case of rational-valued $(\mathbf{A}, \mathbf{b})$, Lemma 4 would not hold. Linear complexity of the reduction could probably be maintained under some additional assumptions, such as prior bounds on the sizes of coordinates of the vertices of P.

2. If the input polytope is in the general form $\{\, \mathbf{x} \in \mathbb{R}^n \mid \mathbf{A}\mathbf{x} \leq \mathbf{b} \,\}$, it can be transformed to the form (4) by adding slack variables and translating.

4.1 Elementary constructions

The output min-sum problem will be constructed from small building blocks, which implement certain simple operations on unary pseudomarginals. We call these blocks *elementary constructions*. An elementary construction is a min-sum problem with $|K| = 3$ labels, zero unary costs $g_u(k) = 0$, binary costs $g_{uv}(k,\ell) \in \{0,1\}$, and optimal value $\min_{\boldsymbol{\mu}\in\Lambda}\langle \mathbf{g}, \boldsymbol{\mu}\rangle = 0$. It follows that $\boldsymbol{\mu} \in \Lambda$ is optimal to the LP relaxation if and only if

$$g_{uv}(k,\ell)\,\mu_{uv}(k,\ell) = 0, \quad \{u,v\} \in E;\ k,\ell \in K. \tag{11}$$

We will define elementary constructions by diagrams such as in Figure 1, in which we draw only edges with costs $g_{uv}(k,\ell) = 1$. Edges with costs $g_{uv}(k,\ell) = 0$ are not drawn. We will use the following elementary constructions (see Figure 2):

COPY enforces equality of two unary pseudomarginals a, d in two objects while imposing no other constraints on b, c, e, f. Precisely, given any feasible unary pseudomarginals a, b, c, d, e, f, there exist feasible binary pseudomarginals satisfying (11) if and only if $a = d$.

ADDITION adds two unary pseudomarginals a, b in one object and represents the result as a unary pseudomarginal $c = a+b$ in another object. No other constraints are imposed on the remaining unary pseudomarginals.

EQUALITY enforces equality of two unary pseudomarginals a, b in a single object, introducing two auxiliary objects. No other constraints are imposed on the remaining unary pseudomarginals. In the sequel, this construction will be abbreviated by omitting the two auxiliary objects and writing the equality sign between the two nodes, as shown in Figure 2(d).

POWERS creates the sequence of unary pseudomarginals with values $2^i a$ for $i = 0, \dots, d$, each in a separate object. We call d the *depth* of the pyramid.

NEGPOWERS is similar to POWERS but constructs values 2^{-i} for $i = 0, \dots, d$.

Figure 3 shows an example of how the elementary constructions can be combined. The edge colors distinguish different elementary constructions. By summing selected bits from NEGPOWERS, the number $\frac{5}{8}$ is constructed. The example can be easily generalized to construct the value $2^{-d}k$ for any $d, k \in \mathbb{N}$ such that $2^{-d}k \le 1$.

4.2 The algorithm

Now we are ready to describe the encoding algorithm. The input of the algorithm is a set of equalities (7). Its output will be a min-sum problem $(V, E, K, \mathbf{g})$ with $|K| = 3$ labels and costs $g_u(k) = 0$, $g_{uv}(k,\ell) \in \{0,1\}$. We will number labels and objects by integers, $K = \{1,2,3\}$ and $V = \{1, \dots, |V|\}$.

The algorithm is initialized as follows:

1.1. For each variable x_j in (4), introduce a new object j into V. The variable x_j will be represented (up to scale) by pseudomarginal $\mu_j(1)$.

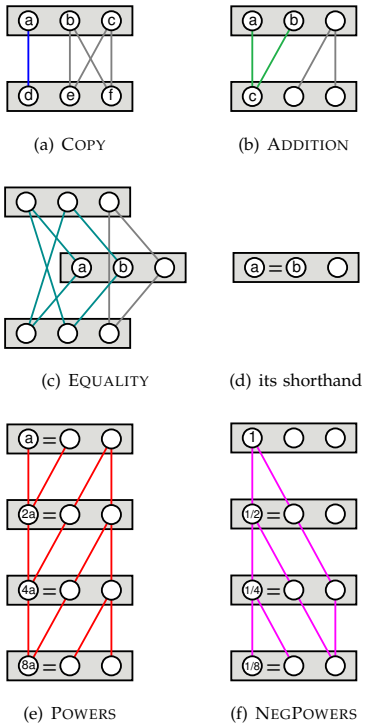


Fig. 2. Elementary constructions.

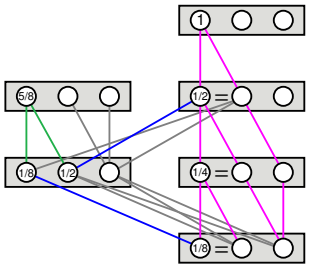


Fig. 3. Construction of the number $\frac{5}{8}$.

1.2. For each such object j, build POWERS to the depth $d_j = \lfloor \log_2 \max_{i=1}^m |a_{ij}| \rfloor$ based on label 1. This yields the sequence of numbers $2^i \mu_j(1)$ for $i = 0, \dots, d_j$.

1.3. Build NEGPOWERS to the depth $d = \lceil \log_2 N \rceil$.

Then the algorithm proceeds by encoding each equation (7). The i-th equation is encoded as follows:

2.1. Construct pseudomarginals with non-zero values $|a_{ij}|x_j$, $j = 1, \dots, n$, by summing selected values from POWERS built in Step 1.2, similarly as in Figure 3. Note that the depths d_j are large enough to make this possible.

2.2. Construct a pseudomarginal with value $2^{-d}b_i$ by summing selected bits from NEGPOWERS built in

Step 1.3, similarly as in Figure 3. The value $2^{-d}b_i$ represents b_i, which sets the scale (mentioned in Theorem 1) between the input and output polytope to 2^{-d}. Note, the depth d is large enough to ensure that all pseudomarginals are bounded by 1.

2.3. Sum all the terms on each side of the equation by repetitively applying ADDITION and COPY.

2.4. Apply COPY to enforce equality of the two sides of the equation.

Figure 4 shows the output min-sum problem for an example polytope P. By construction, the resulting min-sum problem encodes the input polytope as follows:

- If $P = \emptyset$ then $\min_{\boldsymbol{\mu}\in\Lambda}\langle \mathbf{g}, \boldsymbol{\mu}\rangle > 0$.
- If $P \neq \emptyset$ then $\min_{\boldsymbol{\mu}\in\Lambda}\langle \mathbf{g}, \boldsymbol{\mu}\rangle = 0$ and

$$P = \pi(\Lambda^*(\mathbf{g})) \tag{12}$$

where $\pi\colon \mathbb{R}^I \to \mathbb{R}^n$ is the scaled coordinate-erasing projection given by $\pi(\boldsymbol{\mu}) = 2^d(\mu_1(1), \dots, \mu_n(1))$.

Let us make some remarks on this construction. The output min-sum problem has costs $\mathbf{g} \in \{0,1\}^I$ but we could clearly use $\mathbf{g} \in \{0,\infty\}^I$ without affecting the result. The min-sum problem with costs in $\{0,\infty\}$ is well-known as the *constraint satisfaction problem* (CSP). An instance of CSP is *arc consistent* [1] if

$$\min_{\ell\in K} g_{uv}(k,\ell) = g_u(k), \quad u \in V,\ v \in N_u,\ k \in K. \tag{13}$$

Our constructed min-sum problem is arc consistent.

Solving the LP relaxation of the problem $(V, E, K, \mathbf{g})$ decides whether $P \neq \emptyset$ and if so, it finds $\mathbf{x} \in P$. But this in fact means it solves the system $\{\, \mathbf{A}\mathbf{x} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0}\,\}$. Thus, we have the following side-result.

Theorem 6. *Solving any system of linear inequalities reduces in linear time to the LP relaxation of an arc consistent min-sum problem with 3 labels and costs in* $\{0,\infty\}$.

4.3 The complexity of encoding

Let us show that the running time of the algorithm in §4.2 is linear in the size of P, i.e., in the size of the matrix (5). It is usual (see e.g. [10]) to define the description size of a matrix as the number of bits needed to encode all its entries in binary. Since an integer $a \in \mathbb{Z}$ needs at least $\log_2(|a|+1)$ bits to encode, the number

$$L_1 = \sum_{j=1}^{n+1}\sum_{i=1}^{m} \log_2(|\bar{a}_{ij}|+1) \tag{14}$$

is a lower bound on the size of $\bar{\mathbf{A}}$. Now it suffices to show that the running time is $\mathcal{O}(L_1)$ because then it will clearly be linear also in the true size of P.

Note that zero entries $\bar{a}_{ij} = 0$ do not contribute to L_1. Thus L_1 is a lower bound on a *sparse* representation of $\bar{\mathbf{A}}$, in which only non-zero entries are stored.

The running time of the algorithm is obviously[3] linear in $|E|$. Object pairs are created only when an object is created and the number of object pairs added with one object is bounded by a constant, hence $|E| = \mathcal{O}(|V|)$. So it suffices to show that $|V| = \mathcal{O}(L_1)$.

On initialization, the algorithm creates $\sum_{j=1}^{n}(d_j+1)$ objects in Step 1.2 and $d+1$ objects in Step 1.3. It is easy to verify that both these numbers are $\mathcal{O}(L_1)$. To show that $d+1 = \mathcal{O}(L_1)$, one needs to show (referring to (10)) that $\log_2 M = \mathcal{O}(L_1)$ and $\log_2 \max_i \sum_j |a_{ij}| = \mathcal{O}(L_1)$.

For illustration, we only prove $\log_2 M = \mathcal{O}(L_1)$ and leave the rest up to the reader. For every j, we have

$$\sum_{i=1}^{m} |\bar{a}_{ij}| \leq \prod_{i=1}^{m} (|\bar{a}_{ij}|+1)$$

because multiplying out the left-hand side yields the right-hand side plus additional non-negative terms. Taking logarithm and summing over j yields

$$\log_2 M = \sum_{j=1}^{n+1}\log_2 \sum_{i=1}^{m} |\bar{a}_{ij}| \leq \sum_{j=1}^{n+1}\sum_{i=1}^{m} \log_2(|\bar{a}_{ij}|+1) = L_1.$$

Finally, encoding one equality (7) adds at most as many objects as there are bits in the binary representation of all its coefficients. Thus, the number of objects added to encode all equalities (7) is $\mathcal{O}(L_1)$.

5 ENCODING A LINEAR PROGRAM

We now show how to reduce any linear program to linear optimization over a local marginal polytope. By saying that problem A can be reduced to problem B we mean there is an oracle to solve problem B which takes constant time and which can be called (possibly repeatedly) to solve problem A (this is known as *Turing reduction* [15]).

We assume the input linear program in the form

$$P^*(\mathbf{c}) = \operatorname*{argmin}_{\mathbf{x}\in P} \langle \mathbf{c}, \mathbf{x}\rangle \tag{15}$$

where $\mathbf{c} = (c_1, \dots, c_n) \in \mathbb{Z}^n$. Since the encoding in §4 can be applied only to a bounded polyhedron but the LP (15) can be unbounded, we first need a lemma.

Lemma 7. *Every linear program can be reduced in linear time to a linear program over a bounded polyhedron.*

Proof: Denote $H(\alpha) = \{\, \mathbf{x} \in \mathbb{R}^n \mid \langle \mathbf{1}, \mathbf{x}\rangle \leq \alpha\,\}$. By Lemma 4, all vertices of P are contained in the halfspace $H(nM)$. Clearly,

$$\min_{\mathbf{x}\in P\cap H(nM)} \langle \mathbf{c}, \mathbf{x}\rangle \ \geq\ \min_{\mathbf{x}\in P\cap H(2nM)} \langle \mathbf{c}, \mathbf{x}\rangle. \tag{16}$$

Each side of (16) is a linear program over a bounded polyhedron. Inequality (16) is tight if and only if (15)

3. The only thing that may not be obvious is how to multiply large integers a, b in linear time. But this issue can be avoided by instead computing $p(a,b) = 2^{\lceil \log_2 a\rceil + \lceil \log_2 b\rceil}$, which can be done in linear time using bitwise operations. Since $ab \leq p(a,b) \leq (2a)(2b)$, the bounds like M become larger but this does not affect the overall complexity.

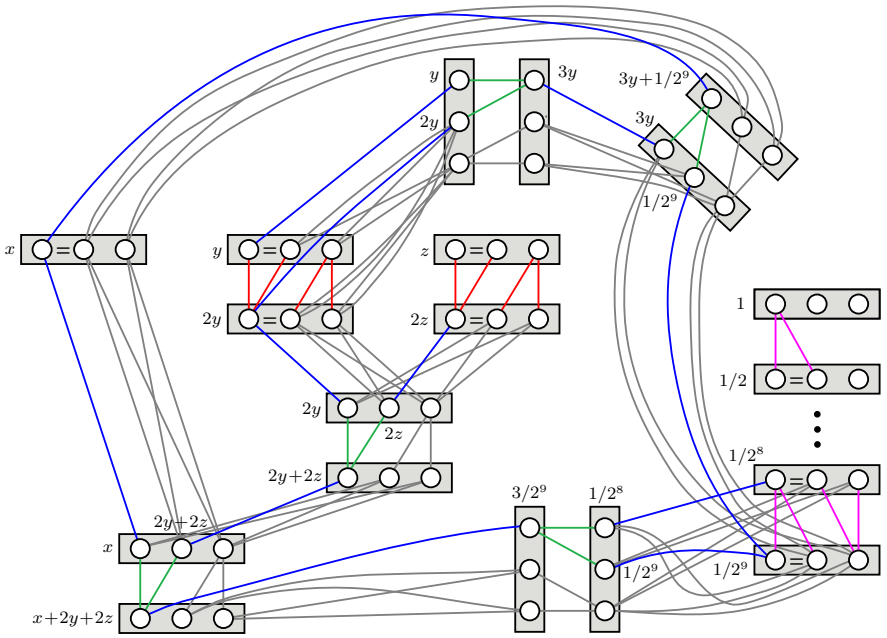


Fig. 4. The output min-sum problem for the polytope $P = \{\,(x, y, z) \mid x + 2y + 2z = 3;\ -x + 3y = -1;\ x, y, z \geq 0\,\}$.

is bounded, in which case (16) has the same optimum as (15). The linear programs (16) are infeasible if and only if (15) is infeasible.

The description size of numbers nM and $2nM$ is $\mathcal{O}(L_1)$, thus the reduction is done in linear time. □

By Lemma 7, we further assume that P is bounded. We also assume that $P \neq \emptyset$ because $P = \emptyset$ is indicated by $\min_{\boldsymbol{\mu} \in \Lambda} \langle \mathbf{g}', \boldsymbol{\mu} \rangle > 0$.

By Theorem 1, optimizing a linear function over P can be reduced in linear time to optimizing a linear function over a face of Λ. Given an oracle to optimize a linear function over Λ, it may seem unclear how to optimize a linear function over a *face* of Λ. This can be done by setting non-zero binary costs to a large constant.

Precisely, let $(V, E, K, \mathbf{g}')$ be the min-sum problem that encodes P, constructed in §4. Define $\mathbf{g} \in \overline{\mathbb{R}}^I$ by

$$g_i(k) = \begin{cases} c_i & \text{if } k = 1 \text{ and } i \in \{1, \dots, n\}, \\ 0 & \text{otherwise,} \end{cases} \tag{17a}$$

$$g_{ij}(k, \ell) = \begin{cases} 0 & \text{if } g'_{ij}(k, \ell) = 0, \\ g_\infty & \text{if } g'_{ij}(k, \ell) = 1. \end{cases} \tag{17b}$$

By (17a), we have $\langle \mathbf{g}, \boldsymbol{\mu} \rangle = \langle \mathbf{c}, \pi(\boldsymbol{\mu}) \rangle$ for every $\boldsymbol{\mu}$. The constant g_∞ in (17b) is large enough to ensure that every $\boldsymbol{\mu} \in \Lambda^*(\mathbf{g})$ satisfies (11). It follows that

$$P^*(\mathbf{c}) = \pi(\Lambda^*(\mathbf{g})). \tag{18}$$

It remains to choose g_∞. The situation is different depending on whether or not we are allowed to use infinite costs. If infinite costs are allowed, we simply set $g_\infty = \infty$. This proves Theorem 2.

If infinite costs are not allowed, g_∞ must be large enough but finite. Unfortunately, manipulation with these large numbers increases the complexity of the reduction. This is given by Theorem 9. To prove it, we first need a lemma, which refines Lemma 4 for the special case of the local marginal polytope.

Lemma 8. *Let $\boldsymbol{\mu} \in \mathbb{R}^I$ be a vertex of the local marginal polytope defined by (V, E, K) with $|K| = 3$. Then each component μ of $\boldsymbol{\mu}$ satisfies $\mu = 0$ or $\mu \geq M_\Lambda^{-1}$ where*

$$M_\Lambda = 2^{|V| + 6|E|}. \tag{19}$$

Proof: Write the local marginal polytope in the form (4), i.e., constraints (2) read $\mathbf{A}\mathbf{x} = \mathbf{b}$. Matrix $\mathbf{A}$ has $|V| + 6|E|$ rows and $3|V| + 9|E|$ columns. Each row of matrix $[\mathbf{A} \,|\, \mathbf{b}]$ has exactly 4 non-zeros, each of them in $\{-1, 1\}$. By Hadamard's inequality, in (9) we have $|\det \mathbf{A}'_j|, |\det \mathbf{A}'| \leq M_\Lambda$. □

Theorem 9. *Every linear program (15) can be reduced to a linear optimization (allowing only finite costs) over a local marginal polytope with 3 labels. The size of the output and the reduction time are $\mathcal{O}(L_1(L_1 + L_2))$ where L_2 is the description size of $\mathbf{c}$.*

Proof: Choose $g_\infty = 1 + M_\Lambda(C_2 - C_1)$ where

$$C_1 = \sum_{i=1}^{n} \min\{0, c_i\}, \qquad C_2 = \sum_{i=1}^{n} \max\{0, c_i\}.$$

We show that now every $\boldsymbol{\mu} \in \Lambda^*(\mathbf{g})$ satisfies (11). It suffices to show this only for vertices of $\Lambda^*(\mathbf{g})$ because taking convex combinations of vertices preserves (11).

Since $\boldsymbol{\mu} \in [0,1]^I$, the contribution of the unary terms to $\langle \mathbf{g}, \boldsymbol{\mu} \rangle$ is in the interval $[C_1, C_2]$. Since $P \neq \emptyset$, we have $\min_{\boldsymbol{\mu} \in \Lambda} \langle \mathbf{g}', \boldsymbol{\mu} \rangle = 0$ and therefore $\min_{\boldsymbol{\mu} \in \Lambda} \langle \mathbf{g}, \boldsymbol{\mu} \rangle \leq C_2$.

Suppose there is a vertex $\boldsymbol{\mu}$ of $\Lambda^*(\mathbf{g})$ and a label pair $\{(u,k),(v,\ell)\}$ such that $g_{uv}(k,\ell) = g_\infty$ and $\mu_{uv}(k,\ell) > 0$. By Lemma 8, we have $\mu_{uv}(k,\ell) \geq M_\Lambda^{-1}$. Thus

$$\min_{\boldsymbol{\mu} \in \Lambda} \langle \mathbf{g}, \boldsymbol{\mu} \rangle \geq g_\infty M_\Lambda^{-1} + C_1 > C_2$$

which is a contradiction.

Let us prove the claimed complexity. The binary length of g_∞ is $\mathcal{O}(L_1 + L_2)$. It occurs in $\mathbf{g}$ at $\mathcal{O}(L_1)$ positions, thus the binary length of $\mathbf{g}$ is $\mathcal{O}(L_1(L_1 + L_2))$. □

6 Reduction to Planar Min-Sum

In this section, we show that the reduction can be done even if we require the graph (V, E) of the output min-sum problem to be planar. For that, it suffices to modify the construction in §4.2 to ensure that (V, E) is planar.

Consider a drawing of the graph (V, E) in the plane, in which vertices are distinct points and edges are straight line segments connecting the vertices. We assume w.l.o.g. that no three edges intersect at a common point, except at graph vertices.

The main idea is to replace every edge crossing with an equivalent planar min-sum problem. Consider a pair $\{u,z\},\{v,w\} \in E$ of crossing edges, as shown in Figure 5(a). This pair is replaced by a construction in Figure 5(b). The cost functions $g_{uu'} = g_{vv'}$ copy unary pseudomarginals, i.e., they enforce $\mu_u = \mu_{u'}$ and $\mu_v = \mu_{v'}$. The other cost functions are set as $g_{u''z} = g_{uz}$ and $g_{v''w} = g_{vw}$. Problem H is a planar min-sum problem that enforces unary pseudomarginals in objects u', u'' and v', v'' to be equal, $\mu_{u'} = \mu_{u''}$ and $\mu_{v'} = \mu_{v''}$. This problem can be drawn arbitrarily small so that it is not intersected by any other edges.

Figure 6 shows how the planar min-sum problem H can be designed. We work with halves of unary pseudomarginals, the first two from each object. The order of unary pseudomarginals is changed by swapping neighbors, imitating bubble sort on four elements.

Recall that the (non-planar) min-sum problem constructed in §4.2 has $E = \mathcal{O}(L_1)$ object pairs. Thus, there are $\mathcal{O}(L_1^2)$ edge crossings in this problem, which yields a reduction to a planar min-sum problem (allowing infinite costs) done in time $\mathcal{O}(L_1^2 + L_2)$.

It turns out that a more careful strategy of drawing the graph decreases the bound on edge crossings to $\mathcal{O}(mL_1)$. Before proving this in Theorem 11, we need a lemma.

Suppose we are given numbers $\alpha_1, \ldots, \alpha_p$ and sets $I_1, \ldots, I_q \subseteq \{1, \ldots, p\}$ and we want to compute numbers $\beta_j = \sum_{i \in I_j} \alpha_i$, $j = 1 \ldots, q$. The j-th sum is constructed using a binary tree, T_j, in which every non-leaf vertex is the sum of its children (i.e., every non-leaf vertex with two children is ADDITION and every edge is COPY, as in Figure 3). The leaves of T_j are α_i, $i \in I_j$, and its root is β_j. We refer to this construction as SUMTREES.

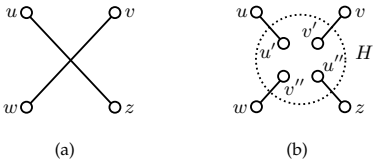


Fig. 5. Eliminating an edge crossing.

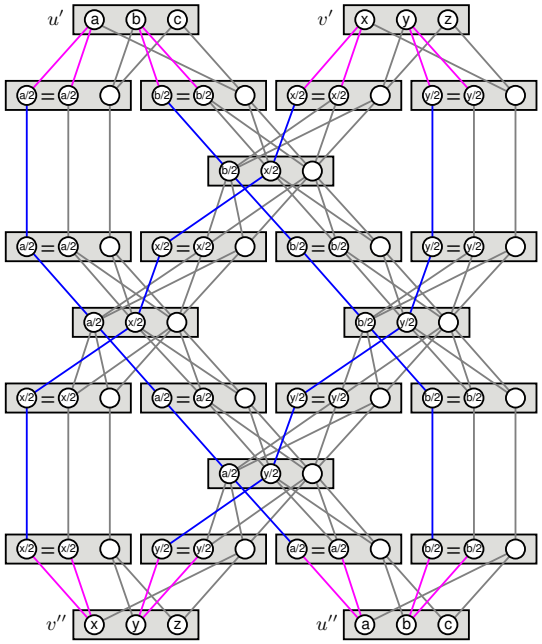
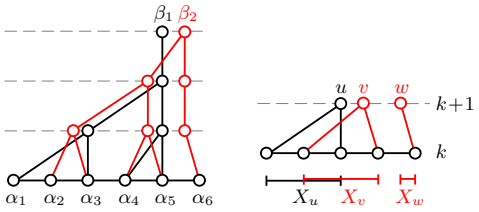


Fig. 6. Planar edge crossing using 3 labels.

Lemma 10. *Let* SUMTREES *be drawn such that the leaves $\alpha_1, \ldots, \alpha_p$ lie on a common horizontal line and their positions on the line are given, and the roots $\beta_1, \ldots, \beta_q$ lie on a different horizontal line and their positions on the line can be arbitrary. Under this constraint,* SUMTREES *can be drawn with $\mathcal{O}(q \sum_{j=1}^q |I_j|)$ edge crossings.*

Proof: The construction is drawn as follows (see Figure 7(a)). Each tree is drawn without edge crossings. In each tree T_j, all the leaves α_i, $i \in I_j$, have the same distance (i.e., the number of edges) to the root β_j. Let the *height* of a tree vertex be defined as its distance to the nearest leaf. The vertical coordinate of every non-root vertex is equal to its height. All the roots $\beta_1, \ldots, \beta_q$ have the same vertical coordinate $h = \lceil \log_2 \max_{j=1}^q |I_j| \rceil$.

Let us focus on tree T_1. It is built in the bottom-up manner. All non-leaf vertices with the same height have two children except the right-most one, which



(a) (b)

Fig. 7. (a) A drawing of SUMTREES for $p = 6$, $q = 2$, $I_1 = \{1, 3, 4, 5\}$, $I_2 = \{2, 3, 4, 5, 6\}$. (b) Crossing edges between two layers.

can have only one child. The horizontal coordinate of a vertex equals the horizontal coordinate of its second child; if there is only one child, it equals the horizontal coordinate of this child. When a layer containing only one vertex has been drawn and its height is less than h, the vertex is linked by a single vertical edge with the layer of height h (thus, this edge can jump over several layers), where it forms the root β_1. Clearly, adding this vertical edge does not affect the overall complexity.

The trees $T_2, \dots, T_q$ are drawn similarly. The only difference is that all non-leaf vertices are shifted to the left by a small offset, to ensure that the non-leaf vertices of all the trees are distinct.

We will show that the number of edge crossings between two trees T_i and T_j is $\mathcal{O}(|I_i| + |I_j|)$. Consider all vertices with heights k and $k+1$ (see Figure 7(b)). For a vertex u with height $k+1$, let $X_u \subset \mathbb{R}$ denote the smallest interval containing the horizontal coordinates of u and its children. Edges going down from u and v to height k can cross each other only if the intervals X_u and X_v intersect. Note that if u and v belong to the same tree, then X_u and X_v are disjoint.

Let $q_{i,k}$ and $q_{j,k}$ be the number of vertices with height k of T_i and T_j, respectively. The number of pairs of intersecting intervals is $\mathcal{O}(q_{i,k}+q_{j,k})$. To see this, observe that if an interval is included in another, then it appears only in one intersecting pair. If all such included intervals are discarded, each interval intersects at most two others. Thus the number of intersections is $\mathcal{O}(q_{i,k} + q_{j,k})$.

It follows that the number of edge crossings between T_i and T_j is $\mathcal{O}(|T_i|+|T_j|)$, where $|T|$ denotes the number of vertices of tree T. But we have $|T_j| = \mathcal{O}(|I_j|)$, because $q_{j,k+1} = \lceil q_{j,k}/2 \rceil$ for every j, k (recall, in every tree the highest non-root layer with a single node is linked with the root layer by a *single* edge).

The total number of crossings in the whole SUMTREES graph is $\sum_{1 \le i \ne j \le q} \mathcal{O}(|I_i| + |I_j|) = \mathcal{O}(q \sum_{j=1}^{q} |I_j|)$. □

Theorem 11. *Every linear program can be reduced in $\mathcal{O}(mL_1 + L_2)$ time to a linear optimization (allowing infinite costs) over a local marginal polytope with 3 labels over a planar graph.*

Proof: It suffices to show how to draw, in the algorithm from §4.2, the graph (V, E) with $\mathcal{O}(mL_1)$ edge crossings. We show this in the rest of the proof.

We start by drawing POWERS for variable x_1 horizontally. Then we draw SUMTREES over the objects of POWERS, with roots being non-zero numbers $|a_{i1}|x_1$, $i = 1, \dots, m$. The i-th tree has $\mathcal{O}(\log_2(|a_{i1}| + 1))$ leaves, therefore, by Lemma 10, this SUMTREES construction has $\mathcal{O}(m \sum_{i=1}^{m} \log_2(|a_{i1}| + 1))$ edge crossings.

This is repeated for the remaining variables $x_2, \dots x_n$, resulting in n independent SUMTREES constructions. The numbers $2^{-d}b_i$, $i = 1, \dots, m$, are constructed similarly, by drawing SUMTREES over NEGPOWERS. The total number of edge crossings is

$$\mathcal{O}\Big(\sum_{j=1}^{n} m \sum_{i=1}^{m} \log_2(|a_{ij}|+1) + m \sum_{i=1}^{m} \log_2(|b_i|+1)\Big) = \mathcal{O}(mL_1).$$

At this stage, we have objects representing all non-zero numbers $|a_{ij}|x_j$ and $2^{-d}b_i$. We assume that the vertical positions of all SUMTREES were such that all these objects lie on a single horizontal line. Now we proceed to sum the terms of each side of each equality (7). This is done by drawing SUMTREES over these objects, with $2m$ roots being the left-hand and right-hand sides of all equalities (7). The tree associated with any side of the i-th equality (7) has $\mathcal{O}(n_i)$ leaves, where n_i is the number of non-zeros in the i-th row of $\mathbf{A}$. Therefore, the number of edge crossings is $\mathcal{O}(m \sum_{i=1}^{m} n_i) = \mathcal{O}(mL_1)$.

At this stage, all objects representing both sides of all equalities (7) lie on a common horizontal line. It remains to join corresponding left- and right-hand sides using COPY. This creates $\mathcal{O}(m^2) \subseteq \mathcal{O}(mL_1)$ edge crossings. □

7 CONSEQUENCES

Let us discuss some consequences of our results.

Most importantly, our results show that solving the LP relaxation of the min-sum problem is comparably hard as solving any LP. This is straightforward if infinite costs are allowed. Then, by Theorem 2, the reduction is done in time $\mathcal{O}(L)$ where $L = L_1 + L_2$, while the best known algorithm [10] for general LP has time complexity[4] $\mathcal{O}(n^{3.5}L^2 \log L \log\log L)$. Finding a very fast algorithm, such as $\mathcal{O}(L^2 \log L)$, to solve the LP relaxation would imply improving the best-known complexity of LP, which is unlikely.

The cases in which the reduction time is polynomial but higher than linear (Theorems 11 and 9) still impose a restriction on possible search for an efficient algorithm to solve the LP relaxation. There are not many principles how to solve the general LP in polynomial time (one is the ellipsoid algorithm), and finding a new such principle is expected to be difficult. Therefore, we should restrict our search to modifying these known principles rather than to discovering a new principle.

4. Note, Karmarkar [10] assumes full encoding of the LP matrix but we allow sparse encoding (see §4.3). To the best of our knowledge, the complexity of solving sparse LPs is largely open [16].

Our results make more precise the known observation that the LP relaxation of the min-sum problem is easier for 2 labels than for the general case. It is known that for 2 labels the LP relaxation reduces in linear time to max-flow [3], [17] and the local marginal polytope has half-integral vertices [11], [23]. For 3 labels, the coordinates of the vertices of local marginal polytopes can have much more general values, as shown in §4.1. Moreover, there is not much difference in complexity between the LP relaxation for 3 labels and for more than 3 labels (allowing infinite costs) because, by Theorem 2, the latter can be reduced to the former in linear time.

Rather than solving directly the LP relaxation (3), it is often more desirable to solve its dual. The dual seeks to maximize a lower bound on (1) by reparameterizations. One class of algorithms to tackle this dual LP converges only to its local minimum, characterized by arc consistency. This class includes popular message passing algorithms [23], [11], [7] and the algorithms [14], [23, §VII], [5]. Theorem 6 has an interesting consequence. Suppose we are given a fixed point of say min-sum diffusion [23] and want to decide whether it is (globally) optimal to the dual LP relaxation and if so, find an optimal solution to the primal LP (3). This problem is equivalent to the LP relaxation of an arc consistent min-sum problem with costs in $\{0, \infty\}$, therefore it is as hard as solving the general system of linear inequalities.

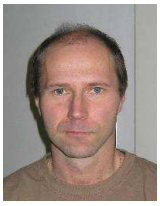
Acknowledgment

Both authors were supported by the Czech Science Foundation grant P202/12/2071. Besides, TW was supported by the European Comission grant FP7-ICT-270138.

REFERENCES

[1] Christian Bessiere. Constraint propagation. In *Handbook of Constraint Programming*, chapter 3. Elsevier, 2006.
[2] Louis J. Billera and A. Sarangarajan. All 0-1 polytopes are traveling salesman polytopes. *Combinatorica*, 16(2):175–188, 1996.
[3] Endre Boros and Peter L. Hammer. Pseudo-Boolean optimization. *Discrete Applied Mathematics*, 123(1-3):155–225, 2002.
[4] Chandra Chekuri, Sanjeev Khanna, Joseph Naor, and Leonid Zosin. Approximation algorithms for the metric labeling problem via a new linear programming formulation. In *Symp. on Discrete Algorithms*, 2001.
[5] M. C. Cooper, S. de Givry, M. Sanchez, T. Schiex, M. Zytnicki, and T. Werner. Soft arc consistency revisited. *Artificial Intelligence*, 174(7-8):449–478, 2010.
[6] Jesús A. De Loera and Shmuel Onn. All linear and integer programs are slim 3-way transportation programs. *SIAM J. on Optimization*, 17(3):806–821, 2006.
[7] Amir Globerson and Tommi Jaakkola. Fixing max-product: Convergent message passing algorithms for MAP LP-relaxations. In *Conf. on Neural Information Processing Systems*, 2008.
[8] Jason K. Johnson, Dmitry M. Malioutov, and Alan S. Willsky. Lagrangian relaxation for MAP estimation in graphical models. In *Allerton Conf. on Communication, Control and Computing*, 2007.
[9] Jörg H. Kappes, Bjoern Andres, Fred A. Hamprecht, Christoph Schnörr, Sebastian Nowozin, Dhruv Batra, Sungwoong Kim, Bernhard X. Kausler, Jan Lellmann, Nikos Komodakis, and Carsten Rother. A comparative study of modern inference techniques for discrete energy minimization problem. In *Conf. on Computer Vision and Pattern Recognition*, 2013.
[10] Narendra Karmarkar. A new polynomial-time algorithm for linear programming. In *ACM Symp. Theory of Computing*, 1984.
[11] Vladimir Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 28(10):1568–1583, 2006.
[12] Nikos Komodakis, Nikos Paragios, and Georgios Tziritas. MRF energy minimization and beyond via dual decomposition. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 33(3):531–552, 2011.
[13] Arie Koster, Stan P.M. van Hoesel, and Antoon W.J. Kolen. The partial constraint satisfaction problem: Facets and lifting theorems. *Operations Research Letters*, 23(3–5):89–97, 1998.
[14] V. K. Koval and Michail I. Schlesinger. Dvumernoe programmirovanie v zadachakh analiza izobrazheniy (Two-dimensional programming in image analysis problems). *USSR Academy of Science, Automatics and Telemechanics*, 8:149–168, 1976. In Russian.
[15] Christos M. Papadimitriou. *Computational complexity*. Addison-Wesley, 1994.
[16] Panos M. Pardalos and Stephen A. Vavasis. Open questions in complexity theory for numerical optimization. *Math. Program.*, 57:337–339, 1992.
[17] Carsten Rother, Vladimir Kolmogorov, Victor S. Lempitsky, and Martin Szummer. Optimizing binary MRFs via extended roof duality. In *Conf. on Computer Vision and Pattern Recognition*, 2007.
[18] Michail I. Shlezinger. Syntactic analysis of two-dimensional visual signals in noisy conditions. *Cybernetics and Systems Analysis*, 12(4):612–628, 1976.
[19] David Sontag, Amir Globerson, and Tommi Jaakkola. Introduction to dual decomposition for inference. In *Optimization for Machine Learning*. MIT Press, 2011.
[20] Johan Thapper and Stanislav Živný. The power of linear programming for valued CSPs. In *Symp. Foundations of Computer Science*. IEEE, 2012.
[21] Stanislav Živný. *The Complexity of Valued Constraint Satisfaction Problems*. Cognitive Technologies. Springer, 2012.
[22] Martin J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305, 2008.
[23] Tomáš Werner. A linear programming approach to max-sum problem: A review. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 29(7):1165–1179, 2007.

Daniel Průša received his PhD degree at Charles University in Prague, in 2005. In 1999-2010, he worked as a software engineer in Sun Microsystems. Since 2010, he works as a researcher at the Center for Machine Perception at the Czech Technical University in Prague. His interests include computational complexity, theory of formal languages, and optimization.



Tomáš Werner received his PhD degree at the Czech Technical University in Prague, in 1999. Since then, he worked as a researcher at the Center for Machine Perception at the same university. The years 2000-2001 he spent as a postdoc in the Visual Geometry Group at the Oxford University, U.K. His interests include multiple view geometry, graphical models, constraint satisfaction, and optimization.

Marginal Consistency: Upper-Bounding Partition Functions over Commutative Semirings

Tomáš Werner

Abstract—Many inference tasks in pattern recognition and artificial intelligence lead to partition functions in which addition and multiplication are abstract binary operations forming a commutative semiring. By generalizing max-sum diffusion (one of convergent message passing algorithms for approximate MAP inference in graphical models), we propose an iterative algorithm to upper bound such partition functions over commutative semirings. The iteration of the algorithm is remarkably simple: change any two factors of the partition function such that their product remains the same and their overlapping marginals become equal. In many commutative semirings, repeating this iteration for different pairs of factors converges to a fixed point when the overlapping marginals of every pair of factors coincide. We call this state marginal consistency. During that, an upper bound on the partition function monotonically decreases. This abstract algorithm unifies several existing algorithms, including max-sum diffusion and basic costraint propagation (or local consistency) algorithms in constraint programming. We further construct a hierarchy of marginal consistencies of increasingly higher levels and show than any such level can be enforced by adding identity factors of higher arity (order). Finally, we discuss instances of the framework for several semirings, including the distributive lattice and the max-sum and sum-product semirings.

Index Terms—partition function, commutative semiring, graphical model, Markov random field, linear programming relaxation, message passing, max-sum diffusion, soft constraint satisfaction, local consistency, constraint propagation

1 Introduction

A *partially separable function* is the product $\prod_{A\in E} f_A(x_A)$ where $E \subseteq 2^V$ is a hypergraph and each *factor* f_A is a function of variables $x_A = (x_i)_{i\in A}$. The sum of the values of this function over all the variables $x_V = (x_i)_{i\in V}$ is the *partition function*

$$\sum_{x_V} \prod_{A\in E} f_A(x_A). \tag{1}$$

E.g., for $V = \{1,2,3,4\}$ and $E = \{\{1,3,4\},\{1,2\},\{2,3\}\}$ the partition function is the number

$$\sum_{x_1,x_2,x_3,x_4} f_{134}(x_1,x_3,x_4) \times f_{12}(x_1,x_2) \times f_{23}(x_2,x_3)$$

where we abbreviated $f_{\{1,3,4\}}$ by f_{134}, etc.

It is known [1], [59], [7], [6], [35], [52] that many inference tasks in pattern recognition and artificial intelligence lead to expressions of the form (1) where $+$ and $\times$ are not the ordinary arithmetic operations but abstract binary operations on some set S such that both operations are associative and commutative and $\times$ distributes over $+$. Such a structure $(S,+,\times)$ is known as the *commutative semiring* [26], [24].

The simplest instance is obtained for the *or-and semiring* $(\{0,1\},\max,\min)$. Here the semiring operations have the meaning of logical disjunction and conjunction and (1) is the decision problem asking whether there is a configuration satisfying all the predicates f_A. This problem is known in computer vision and pattern recognition as the *consistent labeling problem* [27], [28] and in artificial intelligence and constraint programming [55] as the *constraint satisfaction problem* (CSP) [45], [20]. The latter name is more widely used today. Here, the factors f_A are usually called *constraints* and the collection f_A, $A \in E$, a *network of constraints*.

The ordinary 'crisp' CSP has been generalized to handle 'soft' constraints, which can be partially satisfied rather than completely satisfied or completely violated [48]. This leads to optimization problems. One important such formulation is obtained for the *max-min semiring* $([0,1],\max,\min)$. This problem was first proposed in [54] but it is more widely known as the *fuzzy CSP* [17]. Another important formulation is obtained for the *max-sum semirings* $(\mathbb{R},\max,+)$ and $(\mathbb{R}\cup\{-\infty\},\max,+)$ where $+$ is the ordinary addition. In computer vision and pattern recognition, this problem has been called a two-dimensional grammar [61], [58], discrete energy minimization [8], [66], [33], MAP inference in graphical models [69], or the max-sum labeling problem [72]. In constraint programming, it has been called the partial [40], weighted [48] or valued [68] CSP. Several other soft CSP formulations exist [48], [5].

Two abstract frameworks have been proposed in constraint programming to unify various formulations of soft CSPs, [56] and [7], [6]. The latter is strictly more general than the former [5] and closely related to our formulation. The main difference to our formulation is that [56], [7], [6] assume that the semiring addition is idempotent ($a + a = a$ for all $a \in S$) because otherwise (1) might no longer be an optimization problem (such as if it is the ordinary partition function).

For the max-sum semiring, a class of approaches to

• *The author is with the Department of Cybernetics, Czech Technical University, Karlovo náměstí 13, 12135 Praha, Czech Republic.*

problem (1) is based on linear programming relaxation [61], [72], [40], [69], [37], [10], [67] or, equivalently, dual decomposition [32], [39], [64]. To solve this LP relaxation, convergent message-passing algorithms have been proposed [41], [37], [32], [23], [47], [64] that monotonically decrease a convex upper bound on (1) by minimizations over blocks of variables. These algorithms are closely related to one another: they converge in infinite time to a fixed point, which is a local (with respect to block-coordinate moves) optimum of the relaxation. The simplest and oldest of them is *max-sum diffusion*, proposed in 1970's for purely unary constraints but never published [41], and recently revisited in [72], [74].

Convergent message passing algorithms repeat a simple local operation which propagates information through the network and monotonically decreases some quantity. In this respect, they resemble *local consistency* (or *constraint propagation*) algorithms [50], [15], [4], used in constraint programming to prune the search space of the CSP. The most widely known local consistency is *arc consistency* [46], [4, §4]. In computer vision, an algorithm equivalent to enforcing arc consistency was proposed by Waltz [70] and revisited by Rosenfeld et al. [54], who called it *discrete relaxation labeling*. Unlike message passing algorithms, local consistency algorithms in CSP converge in polynomial time, so they can be easily maintained during search.

In constraint programming, the question appeared whether local consistency algorithms can be extended from the ordinary CSP to soft CSPs. This turns out to be straightforward for soft CSPs with idempotent semiring multiplication ($a \times a = a$ for all $a \in S$), such as in the max-min semiring. The resulting algorithms are polynomial and their fixed point does not depend on the order of updates [54], [56], [7], [5], [6]. However, for non-idempotent semiring multiplication (such as in the max-sum semiring), no finite network algorithm has been found that would naturally generalize classical local consistency algorithms [56], [7], [5], [6], [12]. Motivated primarily by efficiency in branch-and-bound search, several finite algorithms have been proposed [13] but they provide weaker bounds than convergent message-passing algorithms.

Contribution. In this paper, we show that max-sum diffusion can be naturally generalized to the abstract commutative semiring. The update of the resulting algorithm is remarkably simple:

> *Change two factors f_A and f_B such that the function $f_A \times f_B$ is preserved and the overlapping marginals of f_A and f_B become equal,*

i.e., change f_A and f_B such that $f_A(x_A) \times f_B(x_B)$ is preserved for all $x_{A \cup B}$ and $\sum_{x_{A \setminus B}} f_A(x_A) = \sum_{x_{B \setminus A}} f_B(x_B)$ for all $x_{A \cap B}$. In many semirings, repeating this operation for different pairs of factors converges to a fixed point. This results in the observation that has never before been clearly formulated:

> *In many commutative semirings, every partially separable function can be reparameterized by local operations to a state when the overlapping marginals of each pair of factors coincide.*

We call this state *marginal consistency* and the algorithm *enforcing marginal consistency*. This terminology agrees with that in constraint programming [4], which distinguishes, e.g., *arc consistency* (a property of a network) and *enforcing arc consistency* (an algorithm to achieve this property). Marginal consistency can be enforced in a number of commutative semirings, including the or-and, max-min, max-sum, and sum-product semiring. As special cases, we obtain basic local consistency algorithms in CSP, including arc consistency.

We further show that $\prod_{A \in E} \sum_{x_A} f_A(x_A)$ is an upper bound on the semiring partition function, with respect to the *canonical order* on the semiring [26]. If the semiring satisfies the Cauchy-Schwarz inequality, the upper bound monotonically decreases during the algorithm.

For the max-sum semiring, it was observed that the basic LP relaxation of (1) [61], [72], [40], [10], [67] can be made tighter at the expense of more computational effort [40], [32], [42], [38], [69], [65], [62], [63], [74]. Some researchers proposed whole hierarchies of increasingly tighter LP relaxations [69, §8.5], [63], [74]. This is similar to using increasingly larger subproblems in dual decomposition [39]. In [74], [19], we constructed such a hierarchy by adding 'dummy' zero constraints of higher arities. Zero constraints can be added incrementally during max-sum diffusion in a dual cutting-plane fashion. We generalize this technique to other semirings, obtaining a hierarchy of marginal consistencies of increasingly higher levels. For the or-and semiring, this hierarchy contains (strong) k-consistency in CSP [21], [4].

Even for idempotent semiring multiplication our algorithm is simpler than local consistency algorithms proposed for soft CSPs [56], [7], [5], [6].

Our framework does not cover *belief propagation* (or the *sum-product algorithm*) [51], [69], which computes, in polynomial time for acyclic networks, the exact partition function and marginals. This algorithm (and its junction-tree version) can be generalized to any commutative semiring [1]. This is straightforward because its update rule uses only the operations of the sum-product semiring. In contrast, the max-sum diffusion update [72, §VI.A] uses not only the operations 'max' and 'sum' but also 'minus' and 'divide by 2', which have no counterparts in some semirings. Another difference to [1] is that our algorithm does not compute marginals even on trees, e.g., no simple way is known to extract max-marginals from a max-sum diffusion fixed point.

2 Preliminaries

In the sequel, sets are denoted by $\{\cdots\}$ and ordered tuples by $(\cdots)$. Real closed, open and semiopen intervals are $[a, b]$, (a, b) and $[a, b)$, respectively. Non-negative and positive reals are $\mathbb{R}_+ = [0, \infty)$ and $\mathbb{R}_{++} = (0, \infty)$,

respectively. The set of all subsets of a set A is denoted by 2^A and the set of all its k-element subsets by $\binom{A}{k}$. Newly defined concepts are typed in **boldface**.

2.1 Commutative Semigroups and Semirings

Definition 1. *A* **commutative semigroup** *is a set S endowed with a binary operation $+$ that is associative and commutative. We denote a commutative semigroup by $(S,+)$.*

Definition 2. *A* **commutative semiring** *is a set S endowed with binary operations $+$ and $\times$ such that $+$ is associative and commutative, $\times$ is associative and commutative, and $\times$ distributes over $+$. We denote it by $(S,+,\times)$.*

A commutative semiring can be seen as two commutative semigroups, $(S,+)$ and $(S,\times)$, coupled by distributivity. A commutative semiring may have an **identity element** 1, satisfying $a\times 1 = a$ for all $a \in S$. If an identity element exists, it is unique. A commutative semiring may have a **zero element** 0, satisfying $a + 0 = a$ and $a \times 0 = 0$ for all $a \in S$. If it exists, it is unique.

We will usually abbreviate $a \times b$ by ab. We define $a^n = a \times \cdots \times a$ (n-times) and $na = a + \cdots + a$ (n-times).

2.2 Functions of Blocks of Variables

Let V be a finite set of **variables**. Each variable $i \in V$ attains **states** $x_i \in X_i$, where X_i is a finite **domain** of the variable. A **joint state** (configuration) of variables $A \subseteq V$ is an element x_A of the Cartesian product $X_A = \prod_{i\in A} X_i$. The order of factors in this Cartesian product is given by some fixed total order on V (e.g., for $V = \{1,\dots,n\}$ we can take the natural arithmetic order).

In the sequel, by the symbol x_A we will always denote a joint state, i.e., the ordered tuple $(x_i)_{i\in A} \in X_A$. Moreover, we adopt the following 'implicit restriction' convention: for $B \subset A$, whenever symbols x_A and x_B occur in the same expression then x_B denotes the restriction of x_A to variables B. This convention is often tacitly used and in fact self-evident: if, e.g., $A = \{1,2,3\}$ and $B = \{1,2\}$, then $x_B = (x_1, x_2)$ is *indeed* the restriction of $x_A = (x_1, x_2, x_3)$ to variables $\{1,2\}$.

For $A \subseteq V$, consider an S-valued function of variables A, i.e., a function $X_A \to S$. We call A the **scope** of the function and $|A|$ its **arity** (often called *order*). We define the following two operations on such functions:

1) The **combination** of functions $\phi\colon X_A \to S$ and $\psi\colon X_B \to S$ is the function

$$\phi \times \psi\colon X_{A\cup B} \to S, \quad (\phi\times\psi)(x_{A\cup B}) = \phi(x_A)\times\psi(x_B).$$

2) The **marginalization** (also known as *projection*) of a function $\phi\colon X_A \to S$ onto variables $B \subseteq A$ (or over variables $A \setminus B$) is the function

$$\phi|_B\colon X_B \to S, \quad \phi|_B(x_B) = \sum_{x_{A\setminus B}} \phi(x_A).$$

Example 1. Let $A = \{1,2,3\}$, $B = \{3,4\}$, $\phi\colon X_A \to S$, $\psi\colon X_B \to S$. The combination of functions ϕ and ψ is the function $(\phi\times\psi)(x_1,x_2,x_3,x_4) = \phi(x_1,x_2,x_3)\times\psi(x_3,x_4)$. The marginalization of function ϕ onto variables $C = \{2,3\}$ is the function $\phi|_C(x_2,x_3) = \sum_{x_1}\phi(x_1,x_2,x_3)$.

For two functions $\phi,\psi\colon X_A \to S$, we will write $\phi = \psi$ to denote that $\phi(x_A) = \psi(x_A)$ for all $x_A \in X_A$.

The operators of combination and marginalization are often explicitly used in constraint programming [7], [6], [48]. The set of functions $X_A \to S$ for all $A \subseteq V$ endowed with combination and marginalization is an example of the *valuation algebra* [60], [35], [36], [52]. We state here three of the axioms of the valuation algebra:

1) Combination is associative and commutative.

2) For $\phi\colon X_A \to S$ and $C \subseteq B \subseteq A$,

$$(\phi|_B)|_C = \phi|_C. \tag{2}$$

3) For $\phi\colon X_A \to S$, $\psi\colon X_B \to S$, and $A \cap B \subseteq C \subseteq A \cup B$,

$$(\phi\times\psi)|_C = \phi|_{A\cap C} \times \psi|_{B\cap C}. \tag{3}$$

2.3 Semiring Partition Function

Let $E \subseteq 2^V$ be a hypergraph over V. Let each hyperedge $A \in E$ be assigned a function $f_A\colon X_A \to S$. The partially separable function $\prod_{A\in E} f_A\colon X_V \to S$ can be seen as the combination of the functions f_A and the partition function (1) is the marginal of this function over all the variables, thus it can be written also as $\big(\prod_{A\in E} f_A\big)\big|_\emptyset$.

We refer to each function f_A as a **factor** and to the collection f_A, $A \in E$, as a **network of functions** or simply a **network**. A network can be seen as a map

$$f\colon X_E \to S$$
$$(A, x_A) \mapsto f_A(x_A)$$

where

$$X_E = \{\, (A,x_A) \mid A \in E,\ x_A \in X_A \,\}$$

is the set of **tuples**. Note the abuse of notation: X_i for $i \in V$, X_A for $A \subseteq V$, and X_E for $E \subseteq 2^V$ denote three different things.

3 Enforcing Marginal Consistency

Here we generalize max-sum diffusion and related concepts to the abstract commutative semiring.

3.1 Equivalent Networks and Reparameterizations

Definition 3. *Let $E, E' \subseteq 2^V$. Networks $f\colon X_E \to S$ and $f'\colon X_{E'} \to S$ are* **equivalent** *if $\prod_{A\in E} f_A = \prod_{A\in E'} f'_A$.*

Note that the operation $+$ does not appear in the definition, thus network equivalence is defined only with respect to the semigroup $(S,\times)$. Equivalent networks have the same set of variables V and domains X_i, $i \in V$, but they can have different hypergraphs and factors. When $E = E'$, the networks differ only in the values of the factors. In this case, we say that f' is a **reparameterization** of f. Deciding whether two given

networks are reparameterizations of each other can be easy or hard, depending on the semigroup $(S, \times)$.

Some reparameterizations are local, in the sense that they are restricted only to a part of the network. The simplest such reparameterization is restricted to a subnetwork containing only two factors, f_A and f_B.

Definition 4. *A* **reparameterization of a pair** $\{f_A, f_B\}$ *is a change of f_A and f_B that preserves the function $f_A \times f_B$. A reparameterization of any single pair of factors of a network is a* **local reparameterization** *of the network.*

Example 2. Let $A = \{1, 2\}$ and $B = \{2, 3\}$. A reparameterization of the pair $\{f_A, f_B\}$ is any change of f_{12} and f_{23} that preserves the value $f_{12}(x_1, x_2) \times f_{23}(x_2, x_3)$ for all $x_1 \in X_1$, $x_2 \in X_2$, $x_3 \in X_3$.

Local reparameterizations allow us to traverse through a class of equivalent networks $X_E \to S$. However, some reparameterizations may not be compositions of local reparameterizations. This depends on the semigroup $(S, \times)$. In §5.1 we shall discuss properties of reparameterizations for several concrete semigroups $(S, \times)$.

3.2 Enforcing Marginal Consistency of a Pair

Definition 5. *A pair $\{f_A, f_B\}$ is* **marginal consistent** *if $f_A|_{A\cap B} = f_B|_{A\cap B}$.*

Example 3. For $A = \{1, 2\}$ and $B = \{2, 3\}$, $f_A|_{A\cap B} = f_B|_{A\cap B}$ reads $\sum_{x_1} f_{12}(x_1, x_2) = \sum_{x_3} f_{23}(x_2, x_3)$ for all $x_2 \in X_2$.

Note that marginal consistency is defined only with respect to the semigroup $(S, +)$, the operation $\times$ does not appear in Definition 5.

Definition 6. Enforcing marginal consistency *of a pair $\{f_A, f_B\}$ is a reparameterization of this pair that makes it marginal consistent.*

Enforcing marginal consistency of a pair $\{f_A, f_B\}$ means replacing this pair with a solution $\{f'_A, f'_B\}$ to the equation system

$$f'_A \times f'_B = f_A \times f_B \tag{4a}$$
$$f'_A|_{A\cap B} = f'_B|_{A\cap B}. \tag{4b}$$

In expanded form, this reads

$$\begin{aligned} f'_A(x_A) \times f'_B(x_B) &= f_A(x_A) \times f_B(x_B) \quad \forall x_{A\cup B} \in X_{A\cup B} \\ \sum_{x_{A\setminus B}} f'_A(x_A) &= \sum_{x_{B\setminus A}} f'_B(x_B) \qquad \forall x_{A\cap B} \in X_{A\cap B}. \end{aligned}$$

Note that the system in fact breaks into several smaller independent systems, one for each $x_{A\cap B}$.

As we are in the abstract commutative semiring, it is not clear how many (if any) solutions system (4) has and how to find them. It would be desirable to characterize semirings in which the system is solvable and to give an algorithm to find all its solutions in any such semiring. We have not been able to do this.

It is easy to obtain a partial solution to (4). Using (3), marginalizing (4a) onto variables $A \cap B$ yields

$$f'_A|_{A\cap B} \times f'_B|_{A\cap B} = f_A|_{A\cap B} \times f_B|_{A\cap B}.$$

Substituting (4b) into this yields

$$(f'_A|_{A\cap B})^2 = (f'_B|_{A\cap B})^2 = f_A|_{A\cap B} \times f_B|_{A\cap B} \tag{5}$$

where, for a function ϕ, we abbreviated $\phi^2 = \phi \times \phi$. Similarly, marginalizing (4a) onto variables A yields

$$f'_A \times f'_B|_{A\cap B} = f_A \times f_B|_{A\cap B}. \tag{6}$$

Equation (5) is solvable if the semiring has a square root. The square root may not be unique, thus (5) can have mutliple solutions. Unfortunately, having $f'_B|_{A\cap B}$ we may not be able to solve (6) for f'_A because the semiring may not have division. In fact, it can happen that (5) is solvable but (4) is not (see Example 16).

We shall see in §5 that in many semirings, system (4) has a solution and this solution is often unique.

3.3 Marginal Consistency Algorithm

We now formulate a simple algorithm that iteratively enforces marginal consistency of different pairs of factors in a network $f\colon X_E \to S$. Let these pairs be given by a set $J \subseteq \{\{A, B\} \mid A, B \in E\} = \binom{E}{2}$, which can be seen as an undirected graph over E. The order of updates is given by an infinite sequence $(\{A_k, B_k\})_{k=1}^{\infty}$ of hyperedge pairs, such that each pair $\{A, B\} \in J$ occurs in the sequence an infinite number of times. We call this sequence the **update schedule**.

Algorithm 1 (Marginal consistency algorithm.)

for $k = 1, \ldots, \infty$ **do**
  Enforce marginal consistency of pair $\{f_{A_k}, f_{B_k}\}$.
end for

It turns out that in many commutative semirings, the algorithm converges to a fixed point when all pairs $\{f_A, f_B\}$ for $\{A, B\} \in J$ are marginal consistent. It would be desirable to characterize commutative semirings in which this fact holds and provide a rigorous proof. This is difficult in full generality and we have not done it[1]. We discuss convergence of the algorithm for a number of concrete semirings in §5.

3.4 Higher Levels of Marginal Consistencies

We say that a network has marginal consistency level $J \subseteq \binom{E}{2}$ if $f_A|_{A\cap B} = f_B|_{A\cap B}$ for all $\{A, B\} \in J$. We now extend this definition to levels higher than $\binom{E}{2}$.

Consider a collection of functions f_A, $A \subseteq V$, i.e., a network over the complete hypergraph 2^V. We say that this network is globally marginal consistent if $f_A = f_V|_A$

1. In fact, we cannot speak about convergence yet because we have not defined a metric or topology on the abstract commutative semiring. Endowing a semiring with a topology has been considered in mathematics [24], [26] but this is out of scope of our paper.

for all $A \subseteq V$. But then we have also $f_A|_{A\cap B} = f_B|_{A\cap B}$ for all $A, B \subseteq V$. This immediately follows from (2) because $f_A|_{A\cap B} = (f_V|_A)|_{A\cap B} = f_V|_{A\cap B} = (f_V|_B)|_{A\cap B} = f_B|_{A\cap B}$. By imposing the constraints $f_A|_{A\cap B} = f_B|_{A\cap B}$ for only a subset $J \subseteq \{\{A, B\} \mid A, B \subseteq V\} = \binom{2^V}{2}$ of all possible pairs $\{A, B\}$, we obtain various levels of marginal consistency, which are necessary (but not sufficient) for global marginal consistency.

When we have a network over a hypergraph $E \subset 2^V$ rather than $E = 2^V$, the problem is that for some pairs $\{A, B\} \in J$, the function f_A or f_B may not be in the network. In that case, we require that these missing functions exist outside of the network. This leads to the following definitions.

Definition 7. *A network $f\colon X_E \to S$ is* **globally marginal consistent** *if there exists a function $f_V\colon X_V \to S$ such that $f_A = f_V|_A$ for every $A \in E$. Here the function f_V can either be in the network ($V \in E$) or not ($V \notin E$).*

Example 4. Let $V = \{1, 2, 3, 4\}$ and $E = \{\{1, 3, 4\}, \{1, 2\}, \{2, 3\}\}$. A network $f\colon X_E \to S$ is globally marginal consistent if there exists a function f_{1234} such that $f_{134} = f_{1234}|_{134}$, $f_{12} = f_{1234}|_{12}$, $f_{23} = f_{1234}|_{23}$.

Definition 8. *A network $f\colon X_E \to S$ has* **marginal consistency level** *$J \subseteq \binom{2^V}{2}$ if there exist functions $f_A\colon X_A \to S$, $A \subseteq V$, $A \notin E$, such that $f_A|_{A\cap B} = f_B|_{A\cap B}$ for all $\{A, B\} \in J$.*

Example 5. A network with $V = \{1, 2, 3, 4\}$ and $E = \{\{1, 2\}, \{1, 3\}, \{1, 4\}, \{3.4\}, \{2, 3\}\}$ has marginal consistency level $J = \{(\{1, 2\}, \{1, 3\}), (\{1, 2\}, \{2, 3\}), (\{1, 3\}, \{2, 3\}), (\{1, 3, 4\}, \{1, 3\}), (\{1, 3, 4\}, \{1, 4\})\}$ if $f_{12}|_1 = f_{13}|_1$, $f_{12}|_2 = f_{23}|_2$, $f_{13}|_3 = f_{23}|_3$ there exists a function f_{134} such that $f_{134}|_{13} = f_{13}$, $f_{134}|_{14} = f_{14}$.

Some levels of marginal consistency are implied by lower levels. E.g., every level is implied by some level $J \subseteq \{\{A, B\} \mid B \subset A \subseteq V\}$ because $f_B|_{A\cap B} = f_B|_{A\cap B}$ is implied by $f_A|_{A\cap B} = f_B|_{A\cap B}$ and $f_B|_{A\cap B} = f_{A\cap B}$.

All possible subsets $J \subseteq \binom{2^V}{2}$ form a partially ordered hierarchy of marginal consistencies. The least element of the hierarchy has level $\emptyset$, the top element has level $\binom{2^V}{2}$. Global marginal consistency has level $\{\{V, A\} \mid A \in E\}$ but, by (2), this already implies the top level $\binom{2^V}{2}$. There are two natural intermediate levels:

1) *hyperedge-to-variable coupling*

$$J = \{\{A, \{i\}\} \mid i \in A \in E\}, \tag{7}$$

2) *hyperedge-to-hyperegde coupling* $J = \binom{E}{2}$.

Algorithm 1 can in general enforce marginal consistency levels not greater than $\binom{E}{2}$. We now describe a simple technique (proposed for max-sum diffusion in [74], [19]) how to enforce higher levels. Suppose our semiring has the identity element 1. We call f_A an **identity factor** if $f_A(x_A) = 1$ for every $x_A \in X_A$ (in short, $f_A = 1$). Suppose we extend E by some $A \notin E$ and set $f_A = 1$. Since $1 \times a = a$ for all $a \in S$, this yields an equivalent network. We call this operation **adding an identity factor** to the network. By Definition 8, adding one or more identity factors (of possibly higher arities) and running Algorithm 1 allows us to enforce an arbitrary level of marginal consistency, at the expense of enlarging the network. We shall see in §5.2.1 that in some semirings this is possible even without enlarging the network.

Remark 1. It might seem that adding an identity factor f_A requires to store $|X_A|$ numbers in memory, which may be prohibitive. But this can be alleviated by performing reparameterizations by 'messages' during Algorithm 1, rather than modifying factors 'in place'. This is common in the max-sum semiring [37], [72], [74], [19] but it is possible also in other semirings [73].

Remark 2. Recall that marginal consistency is defined on the semigroup $(S, +)$, so it can be studied independently on the operation $\times$. In the semigroup $(\mathbb{R}_+, +)$ where $+$ is the ordinary addition, the set of globally marginal consistent networks is (up to normalization conditions $f_A|_\emptyset = 1$, $A \in E$) known as the *marginal polytope* and the set of networks with marginal consistency level (7) as the *local marginal polytope* [69]. If E is acyclic, these polytopes are equal [69, Proposition 4.1]. This suggests a question: does this fact extend to other semigroups? Precisely, is it true that for acyclic networks, marginal consistency level (7) implies global marginal consistency? Though for some semigroups the answer is known, in general the question is open.

4 Upper Bound on Partition Function

Max-sum diffusion monotonically decreases an upper bound on the true max-sum partition function. Unlike the partition function, this bound is tractable to compute. At a fixed point of max-sum diffusion, it often happens that the bound is tight (i.e., equal to (1)). In this section, we generalize these concepts to other semirings.

4.1 Canonical Order on a Commutative Semiring

To formulate the upper bound, we first need to define a suitable partial order on the commutative semiring. The standard way of doing this is as follows [26].

Definition 9. *The* **canonical preorder** *on a commutative semigroup $(S, +)$ is the relation $\leq$ on S defined by*

$$a \leq b \iff (a = b) \text{ or } (\exists c \in S)(a + c = b). \tag{8}$$

Note that the condition $a = b$ is redundant if the semigroup $(S, +)$ has a neutral element 0. The relation $\leq$ is reflexive and transitive, hence a preorder. It naturally extends to the semiring $(S, +, \times)$ as follows.

Theorem 1. *The semiring operations are monotone with respect to $\leq$, i.e., for all $a, b, c \in S$ we have*

$$a \leq b \implies a + c \leq b + c, \quad ac \leq bc. \tag{9}$$

Proof: Suppose $a \leq b = a + d$. Then $b + c = a + d + c \geq a + c$ and $bc = (a + d)c = ac + dc \geq ac$. □

In general, the relation $\leq$ is not antisymmetric, therefore it may not be a partial order. Theorem 2 gives some simple conditions sufficient for $\leq$ to be an order or not.

A binary operation $+$ is **idempotent** if $a + a = a$ for all $a \in S$. It is **selective** [26] (also known as *conservative* [11]) if $a + b \in \{a, b\}$ for all $a, b \in S$. Clearly, any selective operation is idempotent. A commutative semigroup $(S, +)$ is **cancellative** if $a + c = b + c$ implies $a = b$ for all $a, b, c \in S$. Cancellation and idempotency exclude each other (by cancellation, $a + a = a$ implies $a = 0$).

Theorem 2. *Let $\leq$ be the canonical preorder on $(S, +)$.*
1) If $(S, +)$ is a group, then $\leq$ is an equivalence, therefore it is not a partial order.

2) If $+$ is idempotent (i.e., $(S, +)$ is a semilattice), then $\leq$ is a partial order and we have

$$a \leq b \quad \Longleftrightarrow \quad a + b = b. \tag{10}$$

Moreover, $+$ is the least upper bound with respect to $\leq$.

3) If $+$ is selective, then $\leq$ is a total order. Moreover, $+$ is the maximum with respect to $\leq$.

Proof: 1) Suppose $a \leq b = a + c$. Since $(S, +)$ is a group, c has an inverse, therefore $b + (-c) = a \geq b$. This shows that $\leq$ is symmetric.

2) Suppose $a \leq b = a+c$. Then $a+b = a+a+c = a+c = b$, which proves (10). Antisymmetry holds by (10). Proving that $a+b$ is the least upper bound of a, b means proving that $a \leq c$ and $b \leq c$ implies $a+b \leq c$. By (10), this means that $a + c = c$ and $b + c = c$ implies $a + b + c = c$. This is true because $a + b + c = (a + c) + (b + c) = c + c = c$.

3) For any a, b, we have either $a + b = a$ or $a + b = b$. By (10), this means either $b \leq a$ or $a \leq b$. □

If the canonical preorder is antisymmetric, we call it the **canonical order**. We shall see in §5 that this is so in many concrete instances.

4.2 The Bound

Now we can introduce a tractable upper bound on the semiring partition function (1).

Theorem 3. *We have*

$$\sum_{x_V} \prod_{A \in E} f_A(x_A) \leq \prod_{A \in E} \sum_{x_A} f_A(x_A) = \prod_{A \in E} f_A|_\emptyset. \tag{11}$$

Proof: Using distributivity, multiply the factors on the right-hand side. This yields all the terms on the left-hand side plus some additional terms. The inequality follows from (8), where c are the additional terms. □

4.3 The Effect of Enforcing Marginal Consistency

Suppose that enforcing marginal consistency of a pair $\{f_A, f_B\}$ is possible, i.e., there exist $\{f'_A, f'_B\}$ satisfying (4). In this section, we show that, under a certain assumption on the semiring, enforcing marginal consistency of the pair never increases the upper bound (11).

Since enforcing marginal consistency of $\{f_A, f_B\}$ affects only the two factors in the bound corresponding to A and B, we want to show that

$$f'_A|_\emptyset \times f'_B|_\emptyset \leq f_A|_\emptyset \times f_B|_\emptyset. \tag{12}$$

From (4b) and using (2) we have

$$f'_A|_\emptyset = (f'_A|_{A \cap B})|_\emptyset = (f'_B|_{A \cap B})|_\emptyset = f'_B|_\emptyset. \tag{13}$$

Recall that if system (4) has a solution, $f'_A|_{A \cap B} = f'_B|_{A \cap B}$ satisfy (5). Suppose the semiring has a square root. It need not be unique, we only require that *some* unary operation $\sqrt{\cdot}$ exists on S satisfying $(\sqrt{a})^2 = a$ (but not necessarily $\sqrt{a^2} = a$) for all $a \in S$. Then (5) has a solution

$$f'_A|_{A \cap B} = f'_B|_{A \cap B} = \sqrt{f_A|_{A \cap B} \times f_B|_{A \cap B}} \tag{14}$$

where $\sqrt{\phi}$ denotes component-wise application of $\sqrt{\cdot}$ to a function ϕ. Using (13) and (14), inequality (12) reads

$$\big(\sqrt{f_A|_{A \cap B} \times f_B|_{A \cap B}}\big|_\emptyset\big)^2 \leq f_A|_\emptyset \times f_B|_\emptyset.$$

Denoting $x_{A \cap B} = i$, $|X_{A \cap B}| = n$, $f_A|_{A \cap B}(x_{A \cap B}) = a_i$, $f_B|_{A \cap B}(x_{A \cap B}) = b_i$, this can be written as

$$\Big(\sum_{i=1}^n \sqrt{a_i b_i}\Big)^2 \leq \Big(\sum_{i=1}^n a_i\Big)\Big(\sum_{i=1}^n b_i\Big). \tag{15}$$

To summarize, we have the following result.

Theorem 4. *Let $\sqrt{\cdot}$ be a unary operation on S that satisfies $(\sqrt{a})^2 = a$ for all $a \in S$ and (15) for all $a_i, b_i \in S$, $i = 1, \ldots, n$. Then enforcing marginal consistency of any pair of factors (if it is possible) does not increase the upper bound (11).*

Inequality (15) is a form of the *Cauchy-Schwarz inequality* on the semiring. When the square root is unique, we have $(\sqrt{a})^2 = a = \sqrt{a^2}$ for all $a \in S$ and therefore (15) can be written in the more familiar form

$$\langle a, b \rangle^2 \leq \langle a, a \rangle \langle b, b \rangle \tag{16}$$

for all $a, b \in S^n$, where $\langle a, b \rangle = a_1 b_1 + \cdots + a_n b_n$ is the 'inner product' on the semiring. Moreover, it can be verified that (16) is implied by the inequality

$$2ab \leq a^2 + b^2 \tag{17}$$

for all $a, b \in S$ (however, (16) does not imply (17) in some semirings). Let us emphasize that when $\sqrt{a^2} = a$ for all $a \in S$ does not hold, (17) may not imply (15).

Theorem 4 says that, under a reasonable assumption on the semiring, every iteration of Algorithm 1 either decreases the upper bound or keeps it unchanged. Given this result, one might think that the algorithm is nothing more than a (block-)coordinate descent to minimize the upper bound by local reparameterizations. However, this does not fully explain Algorithm 1 because a coordinate descent is expected to *strictly* decrease its objective in every iteration, whereas an iteration of Algorithm 1 can keep the bound unchanged. Yet we cannot omit such iterations because they may modify the network in such a way that some later iterations decrease the bound

strictly. This is very obvious in the or-and semiring but it is true also in other semirings.

Of course, monotonic decrease of the bound during Algorithm 1 is neither sufficient nor necessary for its convergence to a fixed point. Although in many semirings these two properties occur together, there can be exceptions (see §5.5).

4.4 The Effect of Adding Identity Factors

In §3.4 we showed how higher levels of marginal consistency can be achieved by adding identity factors. What effect does this have on the upper bound?

When the operation $+$ is idempotent, adding an identity factor $f_A = 1$ to a network preserves the upper bound because $f_A|_\emptyset = \sum_{x_A} 1 = 1$. Suppose Algorithm 1 is at its fixed point. If we now add one or more identity factors to the network, extend the set J, and run the algorithm again, the upper bound may further decrease. Indeed, this is because the added factors extended the space of reparameterizations that Algorithm 1 can reach by local reparameterizations. Identity factors can be added incrementally during Algorithm 1 in a cutting-plane fashion, similarly as in [74], [19]. This incremental scheme ensures monotonic improvement of the bound.

When $+$ is not idempotent, adding an identity factor may increase the upper bound because $\sum_{x_A} 1 \geq 1$. Therefore, adding identity factors has no obvious advantage in, e.g., the sum-product semiring.

Remark 3. There is another, very obvious technique how to tighten the upper bound arbitrarily at the expense of more computational effort: by merging several factors into one. In the max-sum semiring, this corresponds to using larger subproblems in dual decomposition [32], [39]. E.g., if $\{1,2\}, \{2,3\}, \{1,3\} \in E$, we can merge binary factors f_{12}, f_{23}, f_{13} into the ternary factor $f_{12} \times f_{23} \times f_{13}$. This decreases $|E|$ by two, keeps the network equivalent, and may decrease the upper bound. Subsequently enforcing marginal consistency may improve the bound even further. This technique is not limited to semirings with idempotent addition. However, it is less compatible with the concept of local consistencies (e.g., in the or-and semiring it does not lead to k-consistencies, §5.2.2).

4.5 When is the Bound Tight?

In this section, we discuss two natural conditions on a network under which inequality (11) is tight (i.e., holds with equality). One condition will be given by Definition 10 and the other condition is global marginal consistency (Definition 7).

In the max-sum semiring, inequality (11) is tight if all constraints in the network agree on some common global configuration x_V. In [69, §8.4], this condition has been called *strong tree agreement*. We say a tuple $(A, x_A) \in X_E$ is **active** if $f_A(x_A) = f_A|_\emptyset$. It is known [61], [72], [74], [12] that deciding the condition leads to the CSP formed by the active tuples. The condition can be formulated for any commutative semiring as follows.

Definition 10. *A network $f\colon X_E \to S$ satisfies* **active tuple agreement** *if there exists a configuration $x_V \in X_V$ such that the tuple (A, x_A) is active for every $A \in E$.*

Note the implicit restriction (§2.2) in Definition 10: x_A is a restriction of x_V to variables A.

Example 6. Let $V = \{1,2,3,4\}$ and $E = \{\{1,3,4\}, \{1,2\}, \{2,3\}\}$. A network $f\colon X_E \to S$ satisfies active tuple agreement if there exist $x_1 \in X_1$, $x_2 \in X_2$, $x_3 \in X_3$, $x_4 \in X_4$ such that $f_{134}(x_1,x_3,x_4) = f_{134}|_\emptyset$, $f_{12}(x_1,x_2) = f_{12}|_\emptyset$, $f_{23}(x_2,x_3) = f_{23}|_\emptyset$. Here, e.g., $f_{12}|_\emptyset = \sum_{x_1,x_2} f_{12}(x_1,x_2)$.

Theorem 5. *Active tuple agreement is sufficient for inequality (11) to be tight.*

Proof: The claim follows from the chain

$$\prod_{A\in E} f_A(x_A) \overset{\text{(a)}}{\leq} \sum_{y_V} \prod_{A\in E} f_A(y_A) \overset{\text{(b)}}{\leq} \prod_{A\in E} f_A|_\emptyset \overset{\text{(c)}}{=} \prod_{A\in E} f_A(x_A)$$

where inequality (a) follows from (8), inequality (b) is (11), and equality (c) holds by the assumption. □

Theorem 6. *If the operation $+$ is selective and the semigroup $(S,\times)$ is cancellative, active tuple agreement is necessary for inequality (11) to be tight.*

Proof: First observe that by (9) and by cancellation, $a < b$ and $a' \leq b$ implies $aa' < bb'$ for every $a,b,a',b' \in S$. Suppose that for every $x_V \in X_V$ there exists some $A \in E$ such that $f_A(x_A) < f_A|_\emptyset$. Using (11), this implies $\prod_{A\in E} f_A(x_A) < \prod_{A\in E} f_A|_\emptyset$. Since $+$ is selective, this implies that inequality (11) is strict. □

Let us turn to the second condition, global marginal consistency (Definition 7).

Theorem 7. *If the operation $+$ is selective or the operation $\times$ is idempotent, global marginal consistency is sufficient for inequality (11) to be tight.*

Proof: Suppose a network f is globally marginal consistent, i.e., there is a function f_V such that $f_A = f_V|_A$ for every $A \in E$. Then inequality (11) reads

$$\sum_{x_V} \prod_{A\in E} f_V|_A(x_A) \leq \prod_{A\in E} \sum_{x_A} f_V|_A(x_A). \tag{18}$$

We have

$$\prod_{A\in E} \sum_{x_A} f_V|_A(x_A) = \prod_{A\in E} \sum_{x_V} f_V(x_V) = \Big[\sum_{x_V} f_V(x_V)\Big]^{|E|}.$$

By (8), we have $f_V(x_V) \leq f_V|_A(x_A)$ for every $x_V \in X_V$ and $A \subseteq V$. This simply says that a function cannot be greater than its marginal. Therefore,

$$\sum_{x_V} \prod_{A\in E} f_V|_A(x_A) \geq \sum_{x_V} \prod_{A\in E} f_V(x_V) = \sum_{x_V} \big[f_V(x_V)^{|E|}\big].$$

If the operation $\times$ is idempotent, then for any $n \in \mathbb{N}$ and $a \in S$ we have $a^n = a$. If $+$ is selective, it is easy to show that for any $n \in \mathbb{N}$ and any $a_1,\dots,a_n \in S$ we have $\big(\sum_i a_i\big)^n = \sum_i (a_i)^n$. In both cases, we have

$\sum_{x_V} \left[f_V(x_V)^{|E|}\right] = \left[\sum_{x_V} f_V(x_V)\right]^{|E|}$. Combining this with (18) yields that inequality (18) is tight. □

We now compare the strength of active tuple agreement and global marginal consistency.

Theorem 8. *If the operation $+$ is selective, global marginal consistency implies active tuple agreement.*

Proof: By global marginal consistency, there is f_V such that $f_A = f_V|_A$ for all $A \in E$. Take any x_V such that $f_V(x_V) = f_V|_\emptyset$. Such x_V exists because $+$ is selective. We have $f_A(x_A) = f_V|_A(x_A) = f_V|_\emptyset$ (note the implicit restriction: x_A is the restriction of x_V). By (2), $f_A|_\emptyset = (f_V|_A)|_\emptyset = f_V|_\emptyset$. We conclude that $f_A(x_A) = f_A|_\emptyset$. □

When $+$ is selective, for every $a_1, \dots, a_n \in S$ there is some j such that $a_j = \sum_{i=1}^n a_i$. However, such j may not exist when $+$ is not selective. In that case, active tuple agreement is not likely to hold because there may be some $A \in E$ such that no tuple (A, x_A) is active.

On the other hand, it can happen that active tuple agreement does not hold but inequality (11) is tight. For a simple example, take a network with a single unary factor, i.e., $|V| = 1$ and $E = \{V\}$. Trivially, any such network is globally marginal consistent. Let $+$ not be selective and $\times$ be idempotent (as in Example 11). Then constraint agreement may not hold but, by Theorem 7, inequality (11) is tight.

5 Instances of the Framework

Let us now discuss concrete instances of our framework. Since the properties of reparameterizations do not depend on the operation $+$, we find it useful to first discuss reparameterizations in concrete commutative semigroups $(S, \times)$. Then we turn to enforcing marginal consistency in concrete commutative semirings.

Further in §5, symbols $+, \times, 0, 1, \sqrt{\cdot}$ will have their ordinary (non-semiring) meaning. We will distinguish semigroups and semirings only up to isomorphism; e.g., the max-sum semiring $(\mathbb{R}, \max, +)$ and the max-product semiring $(\mathbb{R}_{++}, \max, \times)$ are isomorphic (via logarithm).

5.1 Reparameterizations in Concrete Semigroups

Here we discuss reparameterizations in concrete commutative semigroups. We focus on two questions: (i) How hard is to decide whether two given networks are reparameterizations of each other? (ii) Which reparameterizations are compositions of local reparameterizations? We do not try to answer these questions for any commutative semigroup (which we believe would be difficult) but only for selected semigroups of our interest.

5.1.1 *Semilattice* $(S, \wedge)$

A commutative semigroup $(S, \wedge)$ in which the semigroup operation $\wedge$ is idempotent is a *semilattice* [14]. Equivalently, $\wedge$ is the greatest lower bound with respect to some partial order on S. Examples of semilattices are $(\{0,1\}, \min)$, $([0,1], \min)$, and $(2^U, \cap)$ where U is a set and $\cap$ is the set intersection.

Theorem 9. *In every non-trivial ($|S| > 1$) semilattice, deciding whether two networks are reparameterizations of each other is NP-hard.*

Proof: The claim holds for semilattice $(\{0,1\}, \min)$ because deciding whether a given network is a reparameterization of the zero network $f = 0$ (i.e., $f_A(x_A) = 0$ for all $(A, x_A) \in X_E$) is equivalent to the CSP, hence NP-complete. The general case holds because every non-trivial semilattice has a subsemilattice isomorphic to $(\{0,1\}, \min)$, namely $(\{a, a \wedge b\}, \wedge)$ for any $a, b \in S$. □

In a semilattice, not every reparameterization is as a composition of local reparameterizations. This is shown by the following example.

Example 7. Let $(S, \wedge) = (\{0,1\}, \min)$. Let $V = \{1,2,3\}$, $E = \{\{1,2\}, \{2,3\}, \{1,3\}\}$, $X_1 = X_2 = X_3 = \{1,2\}$, $f_{12}(1,1) = f_{12}(2,2) = f_{23}(1,1) = f_{23}(2,2) = f_{13}(1,2) = f_{13}(2,1) = 1$, $f_{12}(1,2) = f_{12}(2,1) = f_{23}(1,2) = f_{23}(2,1) = f_{13}(1,1) = f_{13}(2,2) = 0$. This network (the 'inconsistent cycle') forms an unsatisfiable CSP, i.e., it is a reparameterization of the zero network. But one easily checks that no local reparameterization is possible.

5.1.2 *Group*

Consider semigroup $(\mathbb{R}, +)$. It is a group, i.e., every element has an inverse. Here, every reparameterization $\{f'_A, f'_B\}$ of a pair $\{f_A, f_B\}$ can be written explicitly as

$$f'_A = f_A + \varphi_{AB} \tag{19a}$$
$$f'_B = f_B - \varphi_{AB} \tag{19b}$$

for some function φ_{AB}: $X_{A \cap B} \to \mathbb{R}$ (a 'message'). It is known [37, Lemma 6.3], [72, Theorem 3] that in $(\mathbb{R}, +)$, every reparameterization is a composition of a finite number of local reparameterizations and that these local reparameterizations (and whether they exist) can be found in polynomial time. Given (19), this shows that every reparameterization is an affine transformation.

Though this result has been proved for networks with unary and binary constraints f_A, it is natural to conjecture (cf. [74, §3.2]) that it extends to any network. Moreover, it is easy to verify that the proofs in [37], [72] apply not only to $(\mathbb{R}, +)$ but to any group.

5.1.3 *Semigroup* $(\mathbb{R} \cup \{-\infty\}, +)$

This semigroup has the sub-semigroup $(\{-\infty, 0\}, +)$ which is a semilattice, thus some reparameterizations are not compositions of local reparameterizations. It has also the sub-semigroup $(\mathbb{R}, +)$, thus some reparameterizations are compositions of a finite number of local reparameterizations, given by (19) for φ_{AB}: $X_{A \cap B} \to \mathbb{R}$. However, a new phenomenon appears [76], [75]: there exist networks f and f' that are reparameterizations of each other and there is an infinite (but no finite) sequence of local reparameterizations of f that converges to f'.

Example 8. Let $V = \{1,2,3\}$, $E = \{\{1,2\},\{2,3\},\{1,3\}\}$, $X_1 = X_2 = X_3 = \{1,2\}$, $f_{12}(1,1) = f_{12}(2,2) = f_{23}(1,1) = f_{23}(2,2) = f_{13}(1,1) = f_{13}(2,2) = f_{13}(1,2) = 0$, $f_{12}(1,2) = f_{12}(2,1) = f_{23}(1,2) = f_{23}(2,1) = f_{13}(2,1) = -\infty$. Consider the sequence of three local reparameterizations (19) where the functions $\varphi_{12,13}$, $\varphi_{23,12}$, $\varphi_{13,23}$ are given by $\varphi_{12,13}(1) = \varphi_{23,12}(1) = \varphi_{13,23}(1) = 1$, $\varphi_{12,13}(2) = \varphi_{23,12}(2) = \varphi_{13,23}(2) = 0$. This sequence decreases the value $f_{13}(1,2)$ by 1 and keeps all other values unchanged. Repeating the sequence therefore converges to $f_{13}(1,2) = -\infty$. However, no finite sequence of local reparameterizations can set $f_{13}(1,2)$ to $-\infty$.

5.2 Distributive Lattice $(S, \vee, \wedge)$

A commutative semiring $(S, \vee, \wedge)$ in which both operations are idempotent and satisfy the absorption law

$$a \vee (a \wedge b) = a = a \wedge (a \vee b) \tag{20}$$

is a *distributive lattice* [14]. Then $\vee$ is the least upper bound and $\wedge$ is the greatest lower bound with respect to the canonical order $\leq$. Equivalence (10) extends to

$$a \wedge b = a \iff a \leq b \iff a \vee b = b. \tag{21}$$

Example 9. The or-and semiring $(\{0,1\}, \max, \min)$ is a distributive lattice.

Example 10. The max-min semiring $([0,1], \max, \min)$ is a distributive lattice.

Example 11. In the or-and and max-min semirings, both operations are selective hence the canonical order is total. In some inference tasks, our preferences may be given by a partial order that is not total. An example is the distributive lattice $(2^U, \cup, \cap)$ for some set U (or a sublattice of this lattice), where $\leq$ is the inclusion relation $\subseteq$ on 2^U. In this case, the value (1) is not exactly what we would like to obtain as the result of inference. We would rather like to find maximal elements of the partially ordered set $\{ \bigwedge_{A \in E} f_A(x_A) \,\big|\, x_V \in X_V \} \subseteq S$, while (1) is the least upper bound of this set. Discussion on how to find maximal elements of this set is out of scope of our paper. Nevertheless, enforcing marginal consistency may decrease the values of some tuples, i.e., simplify the problem.

As $(S, \wedge)$ is a semilattice, reparameterizations are described by §5.1.1. System (4) has the unique solution

$$f'_A = f_A \wedge f_B|_{A \cap B} \tag{22a}$$
$$f'_B = f_B \wedge f_A|_{A \cap B}. \tag{22b}$$

Let us prove (22a). From (6) and (4b) we obtain $f'_A \wedge f'_A|_{A\cap B} = f_A \wedge f_B|_{A\cap B}$. But the absorption law (20) implies $f'_A \wedge f'_A|_{A\cap B} = f'_A$. By symmetry we get (22b).

The update (22) never increases the value of any tuple because, by (8) and (20), we have $a \wedge b \leq a$ for all $a, b \in S$. It follows that the upper bound (11) never increases. This agrees with the fact that the distributive lattice has a unique square root (the solution to $b \wedge b = a$ is $b = a$) which satisfies (17).

The behavior of Algorithm 1 is similar to local consistency algorithms for the crisp CSP and soft CSPs with idempotent semiring multiplication: it converges in finite time and its fixed point is unique. To formulate this statement more precisely, we extend the canonical order $\leq$ from tuples to networks: for f, f': $X_E \to S$ we define $f \leq f'$ if $f_A(x_A) \leq f'_A(x_A)$ for all $(A, x_A) \in X_E$.

Theorem 10. *Algorithm 1 reaches in a finite number of iterations a fixed point. This fixed point is the greatest one among all fixed points that are not greater than the initial network, therefore it is independent of the update schedule.*

Our proof uses the technique proposed in [2]. It is similar to the proof of the well-known Knapster-Tarski fixed point theorem [14].

Proof: Enforcing marginal consistency of a single pair $\{f_A, f_B\}$ is a function that maps a network to a network. We denote this function by p_{AB}. It has the following properties:

$$p_{AB}(p_{AB}(f)) = p_{AB}(f) \qquad \text{(idempotency)}$$
$$p_{AB}(f) \leq f \qquad \text{(intensivity)}$$
$$f \leq f' \implies p_{AB}(f) \leq p_{AB}(f') \qquad \text{(monotonicity)}$$

Note, these are the axioms of a *closure operator* [14].

Algorithm 1 produces a sequence of networks $(f^k)_{k=0}^{\infty}$ defined recurrently by $f^k = p_{A_k B_k}(f^{k-1})$, where f^0 is the initial network and $\{A_k, B_k\}$ is the k-th element of the update schedule.

Any value that any tuple can ever attain during the algorithm belongs to the closure of the set of initial values $\{ f^0_A(x_A) \mid (A, x_A) \in X_E \} \subseteq S$ by the operations $\vee$ and $\wedge$. Due to the lattice structure, this closure has finite size. Therefore, by intensivity, the sequence f^k converges in a finite number of iterations.

Suppose a network f satisfies $f \leq f^0$ and $p_{AB}(f) = f$ for all $A, B \in E$. We will prove by induction that $f \leq f^k$ for any k. Suppose $f \leq f^{k-1}$. By monotonicity,

$$f = p_{A_k B_k}(f) \leq p_{A_k B_k}(f^{k-1}) = f^k.$$

We conclude that the convergence point of the sequence f^k is the greatest common fixed point of all the functions p_{AB}, $\{A, B\} \in J$, among all networks not greater than f^0. □

5.2.1 Adding Identity Factors

In §3.4 and §4.4 we discussed how any level of marginal consistency can be achieved by adding identity factors to the network. Assume that our lattice has an identity element, 1. Distributive lattices have the following advantage, not shared by other semirings.

Theorem 11. *Let $E, F \subseteq 2^V$. Let f: $X_{E\cup F} \to S$ be a network such that $f_A = 1$ for every $A \in F$. Let f': $X_{E\cup F} \to S$ be the fixed point of Algorithm 1 applied to f. Then*

$$\bigwedge_{A \in E \cup F} f'_A = \bigwedge_{A \in E} f'_A. \tag{23}$$

Proof: The claim is proved by the following chain:

$$\bigwedge_{A\in E} f'_A \overset{\text{(a)}}{\leq} \bigwedge_{A\in E} f_A \overset{\text{(b)}}{=} \bigwedge_{A\in E\cup F} f_A \overset{\text{(c)}}{=} \bigwedge_{A\in E\cup F} f'_A \overset{\text{(d)}}{\leq} \bigwedge_{A\in E} f'_A.$$

In (a) and (d), $\leq$ denotes the componentwise partial order. Inequality (a) holds because the update (22) cannot increase the value of any tuple, (b) holds because $f_A = 1$ for every $A \in F$, (c) holds because enforcing marginal consistency is a reparameterization, and (d) holds because $a \wedge b \leq a$ for every $a, b \in S$. □

The theorem says that if we add one or more identity factors to a network and run Algorithm 1, we can then *remove* the updated identity factors from the network because this yields a reparameterization of the initial network. In other words, identity factors can be added only temporarily and thus the level of marginal consistency can be increased without enlarging the network.

This can be understood also as follows. Adding identity factors extends the space of reparameterizations reachable by local reparameterizations. In a distributive lattice, some reparameterizations cannot be composed of local reparameterizations. Adding identity factors, enforcing marginal consistency, and then removing the updated identity factors means performing a reparameterization of the initial network that may not be reachable by local reparameterizations.

In fact, if we *could* minimize the upper bound (11) over *all* reparameterizations, the bound would become tight. Indeed, we can add the identity factor $f_V = 1$ to the network and run Algorithm 1 with $J = \{\, \{V, A\} \mid A \in E \,\}$. By Theorem 7, this makes inequality (11) tight. Now we remove the factor f_V.

5.2.2 Marginal Consistency in CSP

In the or-and semiring $(\{0, 1\}, \max, \min)$, inequality (11) evaluated at the fixed point of Algorithm 1 says the well-known fact that passing a local consistency test is necessary for CSP satisfiability. Here, some levels of marginal consistency coincide with some basic local consistencies in CSP [4]. For a network with unary and binary constraints, local marginal consistency is *arc consistency* [46], [4, §4]. For any network, marginal consistency of level (7) is *generalized arc consistency* [4, §4]. For any network, local marginal consistency is *pairwise consistency* [31], [4, §5.4]. Adding appropriate identity constraints of arity less than or equal to k and enforcing pairwise consistency yields (strong) k*-consistency* [21], [4, §5.2].

5.3 Semirings of Max-Sum Type

5.3.1 Semiring $(\mathbb{R}, \max, +)$

In this semiring, reparameterizations are affine transformation of f (see §5.1.2) and the upper bound (11) is a piecewise-linear convex function of f. Therefore, minimizing the upper bound over all reparameterizations can be formulated as a linear program. This linear program is the natural LP relaxation of problem (1), considered (sometimes in dual form) by many researchers [61], [40], [69], [37], [10], [67].

System (4) has the unique solution

$$f'_A = f_A + (f_B|_{A\cap B} - f_A|_{A\cap B})/2 \tag{24a}$$
$$f'_B = f_B + (f_A|_{A\cap B} - f_B|_{A\cap B})/2, \tag{24b}$$

which immediately follows from (5) and (6). The semiring has a unique square root (the solution to $b + b = a$ is $b = a/2$), which satisfies conditions (17). Algorithm 1 is known as *max-sum diffusion* [41], [72], [74]. It is firmly believed that max-sum diffusion converges to a fixed point in an infinite number of iterations but this was never proved (a slightly weaker form of convergence has been proved in [57]).

For different update schedules, the algorithm can converge to different fixed points with different values of the bound. Therefore, in general it does not find the minimum upper bound over all reparameterization [72], [37]. Precisely, for some networks the bound cannot be decreased by any single local reparameterization but only by multiple local reparameterizations simultaneously. This is a manifestation of the fact that block-coordinate descent may not find the global minimum of a convex non-smooth function [3]. Note the difference to the distributive lattice, where some reparameterizations cannot be composed of local reparameterizations at all.

5.3.2 Semiring $(\mathbb{R} \cup \{-\infty\}, \max, +)$

This semiring, known as the *tropical semiring* [22], is obtained by adding the zero semiring element $-\infty$ to $(\mathbb{R}, \max, +)$. Minimizing the upper bound over local reparameterizations (19) again leads to a linear program. However, by §5.1.3, some reparameterizations are not compositions of local reparameterizations and so this does not yield the minimum upper bound over all reparameterizations. This is not surprising since the semiring has a subsemiring $(\{-\infty, 0\}, \max, +)$ isomorphic to $(\{0, 1\}, \max, \min)$, so this would solve the CSP.

The solution to (4) is unique, given by (24) where the operation '$-$' (minus) is extended from $\mathbb{R}$ to $\mathbb{R} \cup \{-\infty\}$ by defining $a - (-\infty) = -\infty$ for all $a \in \mathbb{R} \cup \{-\infty\}$. The semiring has a unique square root which satisfies (17).

Two stages can be discerned in Algorithm 1. After a finite number of iterations, the set of tuples with values $-\infty$ stops changing, which resolves the 'crisp' part of the problem. Then the algorithm changes only finite tuples, similarly as in the semiring $(\mathbb{R}, \max, +)$.

5.3.3 Max-Sum Semiring with Truncated Addition

This is the semiring $([-1, 0], \max, \otimes)$ where

$$a \otimes b = \max\{-1,\, a + b\}. \tag{25}$$

This semiring is isomorphic to semiring $([0, 1], \max, \otimes')$ where $a \otimes' b = \max\{a + b - 1, 0\}$ is the Łukasziewicz t-norm [34]. The resulting problem (1) is closely related to the k*-weighted CSP* [48, §9.2.2].

The semiring has a square root but it is not unique: $b \otimes b = a$ has always a solution but, e.g., for $a = -1$ the solutions are all $b \in [-1, -\frac{1}{2}]$. However, there exists a square root, $b = a/2$, satisfying (15). With this square root, system (4) has a solution[2]

$$f'_A = \max\{-1,\ f_A + (f_B|_{A\cap B} - f_A|_{A\cap B})/2\} \tag{26a}$$
$$f'_B = \max\{-1,\ f_B + (f_A|_{A\cap B} - f_B|_{A\cap B})/2\}. \tag{26b}$$

In experiments on random networks, we observed that Algorithm 1 always converged to a fixed point.

5.3.4 Max-Sum Semiring with Lexicographic Maximum

This is the semiring $(\mathbb{R}^2, \oplus, \otimes)$ where

$$(a_1, a_2) \oplus (b_1, b_2) = \begin{cases} (b_1, b_2) & \text{if } a_1 < b_1 \\ (a_1, \max\{a_2, b_2\}) & \text{if } a_1 = b_1 \\ (a_1, a_2) & \text{if } a_1 > b_1 \end{cases}$$
$$(a_1, a_2) \otimes (b_1, b_2) = (a_1 + b_1, a_2 + b_2).$$

The operation $\oplus$ is the maximum with respect to the lexicographic order on $\mathbb{R}^2$, which is also the canonical order. The solution to (4) is unique, given by (24) where $(\max, +)$ is replaced by $(\oplus, \otimes)$.

The framework can be easily extended from $\mathbb{R}^2$ to $\mathbb{R}^n$.

5.3.5 Adding Identity Factors

As in §5.2.1, suppose we add identity factors to a network and then apply Algorithm 1 to the resulting network. Unfortunately, nothing like Theorem 11 holds in max-sum semirings, so we now cannot remove the updated identity factors because this might yield a network that is not equivalent to the initial network. Thus, in general, higher levels of marginal consistency can be achieved only at the expense of increasing the number of factors in the network.

This can be alternatively understood as follows. In semiring $(\mathbb{R}, \max, +)$, every reparameterization can be composed of local reparameterizations. Thus, the only way how to extend the space of reparameterizations reachable by local reparameterizations is to add new identity factors. This is in contrast with the distributive lattice (§5.2.1), where it suffices to add identity factors only temporarily.

5.4 Semirings of Sum-Product Type

5.4.1 Semiring $(\mathbb{R}_{++}, +, \times)$

In this semiring, system (4) has the unique solution

$$f'_A = f_A \times \sqrt{f_B|_{A\cap B} \,/\, f_A|_{A\cap B}} \tag{27a}$$
$$f'_B = f_B \times \sqrt{f_A|_{A\cap B} \,/\, f_B|_{A\cap B}}. \tag{27b}$$

The semiring has a unique square root (the only solution to $b^2 = a$ is $b = \sqrt{a}$) which satisfies conditions (17).

2. Note that we cannot write $f'_A = f_A \otimes (f_B|_{A\cap B} - f_A|_{A\cap B})/2$ in (26a), because $(f_B|_{A\cap B} - f_A|_{A\cap B})/2$ may not be in $[-1, 0]$.

The semiring is isomorphic (via logarithm) to semiring $(\mathbb{R}, \oplus, +)$ where

$$a \oplus b = \log(\mathrm{e}^a + \mathrm{e}^b) \tag{28}$$

is the *log-sum-exp operation*. In this semiring, reparameterizations are affine transformations of f and the upper bound (11) is a smooth convex function of f. Algorithm 1 is a block-coordinate descent method to minimize this function over reparameterizations and therefore it converges to its global minimum [3]. It can be shown [76] that the fixed point of the algorithm is unique.

This algorithm is not widely known, it was proposed in [76, §6] and also [47] noticed that max-sum diffusion can be formulated in the sum-product semiring. The minimum upper bound is usually very loose, therefore not useful to approximate (1). Even for acyclic E, the bound is not exact and no finite algorithm is known to compute the fixed point. The algorithm can be seen as a very simple representant of convergent message passing algorithms to minimize convex free energies [30], [71], [29], [69, §7], which can provide better bounds.

5.4.2 Semiring $(\mathbb{R}_+, +, \times)$

This semiring is obtained by adding the zero semiring element 0 to $(\mathbb{R}_{++}, +, \times)$. Since the semigroup $(\mathbb{R}_+, \times)$ is isomorphic to $(\mathbb{R} \cup \{-\infty\}, +)$, reparameterizations are described by §5.1.3. System (4) has a unique solution, given by (27) where we define $a/0 = 0$ for all $a \in \mathbb{R}_+$.

5.4.3 Relation to the Max-Sum Semiring

Define the operation $\oplus_t$ by

$$a \oplus_t b = \frac{(ta) \oplus (tb)}{t} = \frac{\log(\mathrm{e}^{ta} + \mathrm{e}^{tb})}{t}. \tag{29}$$

For every finite t, $(\mathbb{R}, \oplus_t, +)$ is a semiring isomorphic to $(\mathbb{R}, \oplus, +)$. In the limit $t \to \infty$, the operation $\oplus_t$ becomes $\max$. The semiring $(\mathbb{R}, \max, +)$ is no longer isomorphic to $(\mathbb{R}, \oplus, +)$. This process is known as *tropicalization* [22], *dequantization* [44] or the *zero temperature limit* [49].

We said in §5.3.1 that in semiring $(\mathbb{R}, \max, +)$, Algorithm 1 in general does not find the minimum upper bound over all reparameterizations. However, the sequence of fixed points of the algorithm in semirings $(\mathbb{R}, \oplus_t, +)$ for increasing t converges to the *optimal* upper bound [76]. This is the core of proximal projection methods with entropy distances for exactly solving the LP relaxation mentioned in §5.3.1 [53].

5.4.4 Application to CSP

Although in the sum-product semiring the minimum upper bound is usually very loose, in [75] we described an interesting situation when this bound *is* useful. We now revisit this result in the semiring context.

Let $f\colon X_E \to \{0, 1\}$. In semiring $(\{0, 1\}, \max, \min)$, expression (1) equals 1 if the CSP represented by f is satisfiable and 0 if not. In semiring $(\mathbb{R}_+, +, \times)$, expression (1) counts the number of solutions to the CSP

represented by f. This problem is known as the *counting CSP* (#CSP) [9]. Note that $(\{0,1\},+,\times)$ is not a semiring because the set $\{0,1\}$ is not closed under addition.

Let $U^{\text{or,and}} \in \{0,1\}$ be the upper bound (11) at the fixed point of Algorithm 1 applied to the network f in semiring $(\{0,1\},\max,\min)$. Let $U^{+,\times} \in \mathbb{R}_+$ be the upper bound at the fixed point of Algorithm 1 applied to f in semiring $(\mathbb{R}_+,+,\times)$. Clearly, $U^{\text{or,and}} = 1$ is necessary for the CSP represented by f to be satisfiable (see §5.2.2). But $U^{+,\times} \geq 1$ is also necessary for this CSP to be satisfiable, requiring that the CSP has at least one solution.

The update rules in the semirings $(\{0,1\},\max,\min)$ and $(\mathbb{R}_+,+,\times)$ treat zero tuples in the same way: if the former sets some tuple to zero, so does the latter[3]. It follows that $U^{\text{or,and}} = 0$ implies $U^{+,\times} = 0$. Equivalently, $U^{+,\times} > 0$ implies $U^{\text{or,and}} = 1$. However, the opposite implication does not hold: there are CSP instances for which $U^{\text{or,and}} = 1$ and $U^{+,\times} = 0$ [75]. Therefore, Algorithm 1 in semiring $(\mathbb{R}_+,+,\times)$ yields a *strictly stronger* condition necessary for CSP satisfiability than in semiring $(\{0,1\},\max,\min)$.

The algorithm has the drawback that when reparameterizations are represented by messages, some messages can grow unbounded [75]. This is a manifestation of the phenomenon described in Example 8.

5.5 Expectation Semiring

Expectation semirings, introduced in [18], [43], are dissimilar to any semiring we discussed above. An example is the commutative semiring $(\mathbb{R}_{++} \times \mathbb{R}, \oplus, \otimes)$ where

$$(a_1,a_2) \oplus (b_1,b_2) = (a_1+b_1, a_2+b_2)$$
$$(a_1,a_2) \otimes (b_1,b_2) = (a_1b_1, a_1b_2+b_1a_2).$$

As noted in [25, Example 7.3], this semiring can be seen as the semiring of matrices $\begin{bmatrix} a_1 & a_2 \\ 0 & a_1 \end{bmatrix}$ with the usual matrix addition and product. These matrices are positive definite, hence the semiring has multiplicative inverse and unique square root. Therefore, the solution to (4) can be written as (27) where $\times, /, \sqrt{\cdot}$ are matrix operations.

The canonical preorder (8) is not antisymmetric: e.g., we have both $(0,-1) \leq (0,1)$ and $(0,1) \leq (0,-1)$. Therefore the concepts of upper bound and its decrease are meaningless. Despite this, we observed in experiments on random networks that Algorithm 1 always converged to a fixed point.

5.6 Semirings that Do Not Admit Enforcing Marginal Consistency

Not every commutative semiring allows enforcing marginal consistency. For that, system (4) has to be solvable. Furthermore, it is reasonable to require that the canonical preorder $\leq$ is antisymmetric and the semiring satisfies the conditions of Theorem 4. Here we give examples of semirings that violate some of these requirements.

Example 12. In semiring $(\mathbb{R},+,\times)$, the semigroup $(\mathbb{R},+)$ is a group, therefore by Theorem 2 the relation $\leq$ is an equivalence rather than a partial order.

Example 13. In semiring $(\mathbb{N},\max,+)$, system (4) is not always solvable. Indeed, this semiring does not have a square root because $a+a=b$ has no solution for odd b.

Example 14. Semiring $(2^U,\cup,\otimes)$ where 2^U is the set of all subsets of a vector space U, $\cup$ is the set union and $a \otimes b = \{\, x+y \mid x \in a,\, y \in b \,\}$ is the Minkowski set sum. This semiring does not have a square root: e.g., for $U=\mathbb{R}$, there is no $a \subseteq \mathbb{R}$ satisfying $a \otimes a = \{1,2\}$.

Example 15. Semiring $(S,\oplus,\otimes)$ where S is the set of all convex subsets of a vector space U, $a \oplus b = \operatorname{conv}(a \cup b)$, and $a \otimes b = \{\, x+y \mid x \in a,\, y \in b \,\}$. This semiring has a unique square root: the solution to $b \otimes b = a$ is $b = \{\, x/2 \mid x \in a \,\}$. Thus equation (5) always has a unique solution. However, system (4) may not have a solution. This can happen already in the simple case $U=\mathbb{R}$, i.e., the elements of S are intervals. E.g., take $A=\{1,2\}$, $B=\{1\}$, $X_1=\{1\}$, $X_2=\{1,2\}$, $f_1(1)=\{0\}$, $f_{12}(1,1)=\{-2\}$, $f_{12}(1,2)=\{2\}$. The solution to (5) is $f'_1(1)=\operatorname{conv}\{-1,1\}$. But (4a) requires that $f'_{12}(1,1) \otimes f'_1(1) = f_{12}(1,1) \otimes f_1(1) = \{-2\} \otimes \{0\} = \{-2\}$. Clearly, there is no such $f'_{12}(1,1)$.

Example 16. [5, §2.4.5] Semiring $(\mathbb{R}^2,\oplus,\otimes)$ where

$$(a_1,a_2) \oplus (b_1,b_2) = \begin{cases} (b_1,b_2) & \text{if } a_1 < b_1 \\ (a_1, \max\{a_2,b_2\}) & \text{if } a_1 = b_1 \\ (a_1,a_2) & \text{if } a_1 > b_1 \end{cases}$$

$$(a_1,a_2) \otimes (b_1,b_2) = \begin{cases} (a_1,a_2) & \text{if } a_1 < b_1 \\ (a_1,a_2+b_2) & \text{if } a_1 = b_1 \\ (b_1,b_2) & \text{if } a_1 > b_1 \end{cases}$$

The operation $\oplus$ is the same as in §5.3.4. The solution to the equation $(b_1,b_2) \otimes (b_1,b_2) = (a_1,a_2)$ is $(b_1,b_2) = (a_1,a_2/2)$, thus the semiring has a unique square root. Therefore equation (5) always has a unique solution. However, system (4) may not have a solution. This happens, e.g., for $A=\{1,2\}$, $B=\{1\}$, $X_1=\{1\}$, $X_2=\{1,2\}$, $f_1(1)=(0,3)$, $f_{12}(1,1)=(0,2)$, $f_{12}(1,2)=(2,0)$.

For semirings that do not allow enforcing marginal consistency it is an interesting open question whether enforcing marginal consistency only approximately can yield useful upper bounds.

6 Summary

Our goal in this article has been to theoretically investigate the simple algorithm defined in §3, first for the abstract commutative semiring and then for several concrete semirings. Let us review the algorithm once again. We are given a commutative semiring $(S,+,\times)$, a hypergraph $E \subseteq 2^V$, and a collection of functions

3. A similar observation was made for loopy belief propagation [16], showing that it treats zero tuples the same way as the arc consistency algorithm.

$f_A\colon X_A \to S$, $A \in E$. The algorithm visits different pairs $\{f_A, f_B\}$ and changes every pair such that $f_A|_{A\cap B} = f_B|_{A\cap B}$ while preserving the function $f_A \times f_B$. In many semirings, repeating this operation converges to a fixed point when $f_A|_{A\cap B} = f_B|_{A\cap B}$ holds for *each* pair $\{f_A, f_B\}$. Every iteration either decreases or preserves the upper bound $\prod_{A\in E} f_A|_\emptyset$ on the semiring partition function $\left(\prod_{A\in E} f_A\right)\big|_\emptyset$.

We have extended this basic algorithm to achieve higher levels of consistency. This is done by adding identity factors $f_A = 1$ (typically of higher arities) to the network, which preserves the function $\prod_{A\in E} f_A$ but extends the set of reachable reparameterizations and thus may enable further improvement of the bound. This yields a hierarchy of consistencies of increasingly higher levels, necessary for global marginal consistency. For a wide class of semirings, global marginal consistency suffices for the upper bound to be tight.

We have discussed the properties of the algorithm in a number of concrete semirings. In a distributive lattice, the algorithm converges in finite time and its fixed point is unique. An example of a distributive lattice is the or-and semiring, for which various levels of marginal consistency correspond to several classical local consistencies in CSP. In semirings of max-sum type, the algorithm converges in an infinite time and its fixed point depends on the update schedule. It is known as max-sum diffusion. In semirings of sum-product type, the algorithm converges in infinite time and its fixed point is unique. In the log-domain, the algorithm minimizes a smooth convex function by block-coordinate descent. It is a simple example of message passing algorithms with convex free energies.

Finally, let us remark that our article is relevant to two disciplines, pattern recognition and constraint programming, which use different terminology and communicate little with each other. We hope that our paper will narrow this undesirable interdisciplinary gap.

ACKNOWLEDGMENT

The author has been supported by the Czech Science Foundation under project P202/12/2071 and by the European Commission under project FP7-ICT-270138.

REFERENCES

[1] Srinivas M. Aji and Robert J. McEliece. The generalized distributive law. *IEEE Trans. on Information Theory*, 46(2):325–343, 2000.

[2] Krzysztof R. Apt. The rough guide to constraint propagation. In *Conf. on Principles and Practice of Constraint Programming*, pages 1–23. Springer, 1999.

[3] Dimitri P. Bertsekas. *Nonlinear Programming*. Athena Scientific, Belmont, MA, 1999.

[4] Christian Bessiere. *Constraint Propagation*. In Rossi et al. [55], 2006.

[5] 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.

[6] Stefano Bistarelli. *Semirings for Soft Constraint Solving and Programming*. Springer Verlag, 2004.

[7] Stefano Bistarelli, Ugo Montanari, and Francesca Rossi. Semiring-based constraint satisfaction and optimization. *J. ACM*, 44(2):201–236, 1997.

[8] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 23(11):1222–1239, 2001.

[9] Andrei A. Bulatov and Víctor Dalmau. Towards a dichotomy theorem for the counting constraint satisfaction problem. *J. Information and Computation*, 205(5):651–678, 2007.

[10] Chandra Chekuri, Sanjeev Khanna, Joseph Naor, and Leonid Zosin. A linear programming formulation and approximation algorithms for the metric labeling problem. *SIAM Journal on Discrete Mathematics*, 18(3):608–625, 2005.

[11] David Cohen and Peter Jeavons. The complexity of constraint languages. In *Handbook of Constraint Programming*, chapter 8. Elsevier, 2006.

[12] M. C. Cooper, S. de Givry, M. Sanchez, T. Schiex, M. Zytnicki, and T. Werner. Soft arc consistency revisited. *Artificial Intelligence*, 174(7-8):449–478, 2010.

[13] Martin Cooper and Thomas Schiex. Arc consistency for soft constraints. *Artificial Intelligence*, 154(1-2):199–227, 2004.

[14] B. A. Davey and H. A. Priestley. *Introduction to Lattices and Order*. Cambridge University Press, Cambridge, 1990.

[15] Romuald Debruyne and Christian Bessière. Domain filtering consistencies. *Journal of Artificial Intelligence Research*, (14):205–230, 2001.

[16] R. Dechter and R. Mateescu. A simple insight into iterative belief propagation's success. In *Conf. on Uncertainty in Artificial Intelligence*, 2003.

[17] Didier Dubois, Helene Fargier, and Henri Prade. The calculus of fuzzy restrictions as a basis for flexible constraint satisfaction. In *IEEE Conf. on Fuzzy Systems (FUZZ-IEEE)*, 1993.

[18] Jason Eisner. Parameter estimation for probabilistic finite-state transducers. In *Annual Meeting of the Assoc. for Computational Linguistics (ACL)*, pages 1–8, 2002.

[19] V. Franc, S. Sonnenburg, and T. Werner. Cutting plane methods in machine learning. In S. Sra, S. Nowozin, and S. J. Wright, editors, *Optimization for Machine Learning*. MIT Press, 2012.

[20] Eugene Freuder and Alan K. Mackworth. *Constraint satisfaction: An emerging paradigm*. In Rossi et al. [55], 2006.

[21] Eugene C. Freuder. Synthesizing constraint expressions. *Communications of the ACM*, 21(11):958–966, 1978.

[22] Stéphane Gaubert. Methods and applications of (max,+) linear algebra. Technical Report 3088, Institut national de recherche en informatique et en automatique (INRIA), 1997.

[23] Amir Globerson and Tommi Jaakkola. Fixing max-product: Convergent message passing algorithms for MAP LP-relaxations. In *Neural Information Processing Systems*, pages 553–560, 2008.

[24] Jonathan S. Golan. *Semirings and their Applications*. Kluwer Academic, 1999.

[25] Jonathan S. Golan. *Semirings and affine equations over them: theory and applications*. Kluwer Academic, 2003.

[26] Michel Gondran and Michel Minoux. *Graphs, Dioids and Semirings: New Models and Algorithms*. Springer, 2008.

[27] R. M. Haralick and L. G. Shapiro. The consistent labeling problem. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 1(2):173–184, 1979.

[28] Robert M. Haralick and Linda G. Shapiro. *Computer and Robot Vision*. Addison-Wesley Longman Publishing, 1992.

[29] Tamir Hazan and Amnon Shashua. Convergent message-passing algorithms for inference over general graphs with convex free energies. In *Conf. on Uncertainty in Artificial Intelligence*, pages 264–273, 2008.

[30] Tom Heskes. Convexity arguments for efficient minimization of the Bethe and Kikuchi free energies. *Jr. of Artificial Intelligence Research*, 26:153–190, 2006.

[31] P. Janssen, P. Jegou, B. Nouguier, and M.C. Vilarem. A filtering process for general constraint satisfaction problems: Achieving pairwise consistency using an associated binary representation. In *IEEE Workshop on Tools for Artificial Intelligence*, pages 420–427, 1989.

[32] Jason K. Johnson, Dmitry M. Malioutov, and Alan S. Willsky. Lagrangian relaxation for MAP estimation in graphical models. In *Allerton Conf. Communication, Control and Computing*, 2007.

[33] Jörg H. Kappes, Bjoern Andres, Fred A. Hamprecht, Christoph Schnörr, Sebastian Nowozin, Dhruv Batra, Sungwoong Kim, Bernhard X. Kausler, Jan Lellmann, Nikos Komodakis, and Carsten Rother. A comparative study of modern inference techniques for discrete energy minimization problem. In *Conf. Computer Vision and Pattern Recognition*, 2013.

[34] E. P. Klement, R. Mesiar, and E. Pap. *Triangular Norms*. Springer, 2000.

[35] J. Kohlas. *Information Algebras: Generic Structures for Inference*. Springer-Verlag, 2003.

[36] J. Kohlas and N. Wilson. Semiring induced valuation algebras: Exact and approximate local computation algorithms. *Artificial Intelligence*, 172(11):1360–1399, 2008.

[37] Vladimir Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 28(10):1568–1583, 2006.

[38] N. Komodakis and N. Paragios. Beyond loose LP-relaxations: Optimizing MRFs by repairing cycles. In *Eur. Conf. on Computer Vision*, 2008.

[39] N. Komodakis, N. Paragios, and G. Tziritas. MRF energy minimization and beyond via dual decomposition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(3):531–552, 2011.

[40] Arie Koster, C. P. M. van Hoesel, and A. W. J. Kolen. The partial constraint satisfaction problem: Facets and lifting theorems. *Operations Research Letters*, 23(3-5):89–97, 1998.

[41] V. A. Kovalevsky and V. K. Koval. A diffusion algorithm for decreasing the energy of the max-sum labeling problem. Glushkov Institute of Cybernetics, Kiev, USSR. Unpublished, approx. 1975.

[42] Pawan Kumar and Philip H. S. Torr. Efficiently solving convex relaxations for MAP estimation. In *Intl. Conf. on Machine Learning*, pages 680–687. ACM, 2008.

[43] Zhifei Li and Jason Eisner. First- and second-order expectation semirings with applications to minimum-risk training on translation forests. In *Conf. on Empirical Methods in Natural Language Processing (EMNLP)*, pages 40–51, 2009.

[44] Grigori L. Litvinov. Maslov dequantization, idempotent and tropical mathematics: A brief introduction. *Journal of Mathematical Sciences*, 140(3):426–444, 2007.

[45] A. Mackworth. Constraint satisfaction. In *Encyclopaedia of Artificial Intelligence*, pages 285–292. Wiley, 1991.

[46] A. K. Mackworth. Consistency in networks of relations. *Artificial intelligence*, 8(1):65–73, 1977.

[47] Talya Meltzer, Amir Globerson, and Yair Weiss. Convergent message passing algorithms: a unifying view. In *Conf. on Uncertainty in Artificial Intelligence*, pages 393–401, 2009.

[48] Pedro Meseguer, Francesca Rossi, and Thomas Schiex. *Soft Constraints*. In Rossi et al. [55], 2006.

[49] M. Mézard and A. Montanari. *Information, Physics, and Computation*. Oxford University Press, Inc., New York, NY, USA, 2009.

[50] Ugo Montanari. Networks of constraints: Fundamental properties and application to picture processing. *Information Science*, 7:95–132, 1974.

[51] Judea Pearl. *Probabilistic reasoning in intelligent systems: Networks of plausible inference*. Morgan Kaufmann, San Francisco, 1988.

[52] M. Pouly and J. Kohlas. *Generic Inference – A unifying Theory for Automated Reasoning*. John Wiley & Sons, Inc., 2011.

[53] Pradeep Ravikumar, Alekh Agarwal, and Martin J. Wainwright. Message-passing for graph-structured linear programs: Proximal methods and rounding schemes. *Journal of Machine Learning Research*, 11:1043–1080, 2010.

[54] A. Rosenfeld, R. A. Hummel, and S. W. Zucker. Scene labeling by relaxation operations. *IEEE Trans. on Systems, Man, and Cybernetics*, 6(6):420–433, June 1976.

[55] Francesca Rossi, Peter van Beek, and Toby Walsh, editors. *Handbook of Constraint Programming*. Elsevier, 2006.

[56] Thomas Schiex, Hélène Fargier, and Gerard Verfaillie. Valued constraint satisfaction problems: Hard and easy problems. In *Intl. Joint Conf. on Artificial Intelligence)*, pages 631–637, 1995.

[57] M. Schlesinger and K. Antoniuk. Diffusion algorithms and structural recognition optimization problems. *Cybernetics and Systems Analysis*, 47:175–192, 2011.

[58] Michail I. Schlesinger. *Matematicheskie sredstva obrabotki izobrazheniy (Mathematical Tools of Image Processing)*. Naukova Dumka, Kiev, 1989. In Russian.

[59] Michail I. Schlesinger and Boris Flach. Some solvable subclasses of structural recognition problems. In *Czech Pattern Recognition Workshop*. Czech Pattern Recognition Society, 2000.

[60] Prakash P. Shenoy and Glenn Shafer. Axioms for probability and belief-function proagation. In *Conf. on Uncertainty in Artificial Intelligence*, pages 169–198, 1990.

[61] M. I. Shlezinger. Syntactic analysis of two-dimensional visual signals in noisy conditions. *Cybernetics and Systems Analysis*, 12(4):612–628, 1976. Translation from Russian.

[62] D. Sontag, T. Meltzer, A. Globerson, T. Jaakkola, and Y. Weiss. Tightening LP relaxations for MAP using message passing. In *Conf. Uncertainty in Artificial Intelligence*, 2008.

[63] David Sontag. *Approximate Inference in Graphical Models using LP Relaxations*. PhD thesis, Massachusetts Institute of Technology, Dept. of Electrical Eng. and Computer Science, 2010.

[64] David Sontag, Amir Globerson, and Tommi Jaakkola. Introduction to dual decomposition for inference. In Suvrit Sra, Sebastian Nowozin, and Stephen J. Wright, editors, *Optimization for Machine Learning*. MIT Press, 2012.

[65] David Sontag and Tommi Jaakkola. New outer bounds on the marginal polytope. In *Neural Information Processing Systems*, 2007.

[66] Richard Szeliski, Ramin Zabih, Daniel Scharstein, Olga Veksler, Vladimir Kolmogorov, Aseem Agarwala, Marshall Tappen, and Carsten Rother. A comparative study of energy minimization methods for markov random fields with smoothness-based priors. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 30(6):1068–1080, 2008.

[67] Johan Thapper and Stanislav Živný. The power of linear programming for valued CSPs. In *Symp. Foundations of Computer Science*, pages 669–678. IEEE, 2012.

[68] Stanislav Živný. *The Complexity of Valued Constraint Satisfaction Problems*. Cognitive Technologies. Springer, 2012.

[69] Martin J. Wainwright and Michael I. Jordan. Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, 1(1-2):1–305, 2008.

[70] David L. Waltz. Generating semantic descriptions from drawings of scenes with shadows. Technical Report AI271, Massachusetts Institute of Technology, 1972.

[71] Yair Weiss, Chen Yanover, and Talya Meltzer. MAP estimation, linear programming and belief propagation with convex free energies. In *Conf. on Uncertainty in Artificial Intelligence*, 2007.

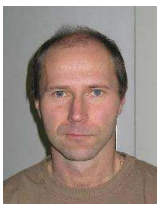
[72] Tomáš Werner. A linear programming approach to max-sum problem: A review. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 29(7):1165–1179, July 2007.

[73] Tomáš Werner. Marginal consistency: Unifying constraint propagation on commutative semirings. In *Intl. Workshop on Preferences and Soft Constraints*, pages 43–57, 2008.

[74] Tomáš Werner. Revisiting the linear programming relaxation approach to Gibbs energy minimization and weighted constraint satisfaction. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 32(8):1474–1488, August 2010.

[75] Tomáš Werner. How to compute primal solution from dual one in MAP inference in MRF? *Control Systems and Computers*, March–April(2):35–45, 2011.

[76] Tomáš Werner and Alexander Shekhovtsov. Unified framework for semiring-based arc consistency and relaxation labeling. In *12th Computer Vision Winter Workshop, St. Lambrecht, Austria*, pages 27–34. Graz University of Technology, February 2007.



Tomáš Werner received his PhD degree at the Czech Technical University in Prague, in 1999. Since then, he worked as a researcher at the Center for Machine Perception at the same university. The years 2000-2001 he spent as a postdoc in the Visual Geometry Group at the Oxford University, U.K. His interests include multiple view geometry, graphical models, optimization, and constraint satisfaction.

Primal View on Belief Propagation

Tomáš Werner

Center for Machine Perception, Czech Technical University
Karlovo náměstí 13, 12135 Prague, Czech Republic

Abstract

It is known that fixed points of loopy belief propagation (BP) correspond to stationary points of the Bethe variational problem, where we minimize the Bethe free energy subject to normalization and marginalization constraints. Unfortunately, this does not entirely explain BP because BP is a dual rather than primal algorithm to solve the Bethe variational problem – beliefs are infeasible before convergence. Thus, we have no better understanding of BP than as an algorithm to seek for a common zero of a system of non-linear functions, not explicitly related to each other. In this theoretical paper, we show that these functions are in fact explicitly related – they are the partial derivatives of a single function of reparameterizations. That means, BP seeks for a stationary point of a single function, without any constraints. This function has a very natural form: it is a linear combination of local log-partition functions, exactly as the Bethe entropy is the same linear combination of local entropies.

1 Introduction

Loopy belief propagation (further only belief propagation, BP) (Pearl, 1988) is a well-known algorithm to approximate marginals and the partition function of the Gibbs probability distribution defined by an undirected graphical model (Markov random field). For acyclic graphs it yields the exact result, for graphs with cycles it often yields surprisingly good approximations. A large body of literature exists on BP and related topics and we refer the reader to the recent survey by (Wainwright & Jordan, 2008).

Unfortunately, BP on cyclic graphs is not guaranteed to converge, which is indeed often observed. A lot of effort has been invested to understanding this phenomenon, see (Wainwright & Jordan, 2008, §4.1.3) for references. Solid ground was provided by (Yedidia et al., 2000; Yedidia et al., 2005) who discovered that BP fixed points coincide with stationary points of the Bethe variational problem, long known in statistical physics. (Heskes, 2003; Heskes, 2006) showed that every stable BP fixed points are local optima (rather than saddle points) of this problem, but not *vice versa*.

The basic operation in the BP algorithm is 'passing a message', which means sending a vector of numbers between a node and an edge of the graph (Pearl, 1988). Messages turned out to be directly related to the Lagrange multipliers of the Bethe variational problem (Yedidia et al., 2000; Yedidia et al., 2005). Later it became clear (Wainwright et al., 2004) that passing a message corresponds to reparameterizing the distribution. In this view, BP tries to reparameterize the distribution so that the corresponding beliefs have consistent marginals.

Though this is generally known, no existing theoretical analysis of BP fully utilizes the interpretation of messages and the Lagrange multipliers of the Bethe variational problem as reparameterizations. In contrast, we incorporate reparameterizations into variational inference and BP in a principled way, which makes the picture more complete and provides a mathematical framework in which we formulate our main result.

The correspondence between BP fixed points and the Bethe variational problem did not entirely explain the BP algorithm itself because BP does not directly solve the Bethe variational problem – beliefs are infeasible to this problem until convergence. Thus, we still have no better understanding of BP than an algorithm to seek for a common zero of a system of non-linear functions, not explicitly related to each other. As our main result, we show that these functions are in fact explicitly related – they are the partial derivatives of a single function of reparameterizations. In other words, BP searches for a stationary point of a single func-

tion, without any constraints. This function has a very natural form: it is a linear combination of local log-partition functions, exactly as the Bethe entropy is the same linear combination of local entropies. We show that BP fixed points are in one-to-one correspondence with stationary points of this function and that all these points are saddles[1].

Several versions of BP and related free energies exist. Originally, BP was formulated for models with pairwise interactions. We formulate our result for the factor-graph BP (Kschischang et al., 2001), which permits interactions of arbitrary order. We currently do not consider more complex versions, cluster variation methods and generalized BP (Yedidia et al., 2005).

2 Exponential families

Here we recall the basics of exponential families of probability distributions, which offer a convenient formalism to reason about graphical models (Wainwright & Jordan, 2008). In §2.3, we incorporate the concept of *reparameterizations* in overcomplete exponential families in a more principled way than other authors – which is important for graphical models, where reparameterizations play a crucial rôle.

Let X and I be finite sets and $\phi\colon X \to \mathbb{R}^I$. The discrete exponential family is a family of probability distributions

$$p(x\,|\,\theta) = \exp[\theta\phi(x) - F(\theta)] \tag{1}$$

instantiated by triplet (X, I, ϕ) and parameterized by vector $\theta \in \mathbb{R}^I$. We understand θ as a row vector and $\phi(x)$ as a column vector, so that $\theta\phi(x) = \sum_{i\in I} \theta_i\phi_i(x)$. The normalization term

$$F(\theta) = \bigoplus_{x\in X} \theta\phi(x) \tag{2}$$

is the *log-partition function* and $a \oplus b = \log(\mathrm{e}^a + \mathrm{e}^b)$ denotes the *log-sum-exp operation*. Operation $\oplus$ is associative and commutative and $+$ distributes over $\oplus$.

We assume in §2.1 and §2.2 that the functions ϕ_i are affinely independent, i.e., they form a *minimal representation* of the family. We relax this later in §2.3.

2.1 Mean parameters

The mean values of functions ϕ_i with respect to distribution (1) are the *mean parameters* of the distribution

$$\mu = \sum_{x\in X} p(x\,|\,\theta)\phi(x) = \frac{\sum_{x\in X} \phi(x)\exp\theta\phi(x)}{\sum_{x\in X}\exp\theta\phi(x)} \tag{3}$$

which is a column vector. The map $\theta \mapsto \mu$ defined by (3) will be denoted $m\colon \mathbb{R}^I \to \mathbb{R}^I$. Parameters θ are *uniquely* determined by μ by solving the equation system $\mu = m(\theta)$. Mean parameters are related with the log-partition function by

$$\frac{\mathrm{d}F(\theta)}{\mathrm{d}\theta} = m(\theta) \tag{4}$$

Let $\phi(X) = \{\, \phi(x) \mid x \in X \,\}$ denote the range of ϕ, a finite set of vectors from $\mathbb{R}^I$. The set of all realizable mean value vectors of ϕ is the convex hull of $\phi(X)$,

$$\operatorname{conv}\phi(X) = \Big\{ \sum_{x\in X} p(x)\phi(x) \;\Big|\; p(x) \geq 0,\ \sum_{x\in X} p(x) = 1 \Big\}$$

where p stands for all possible distributions over X, not necessarily from the family. Every element of $\operatorname{conv}\phi(X)$ with $p(x) > 0$ (i.e., a strictly positive convex combination of ϕ) can be obtained also as the mean of ϕ over a distribution *from* the family – thus, the range $m(\mathbb{R}^I)$ of the map m is the interior of $\operatorname{conv}\phi(X)$.

2.2 Entropy and convex conjugacy

The entropy of distribution (1) as a function of θ equals $F(\theta) - \theta m(\theta)$. Let $H(\mu)$ denote the entropy of the distribution as a function of μ. It is defined implicitly: we first take any θ satisfying $\mu = m(\theta)$ and then let $H(\mu) = F(\theta) - \theta\mu$. The function H is positive and concave and its domain is the interior of $\operatorname{conv}\phi(X)$.

The functions F and $-H$ are related by convex conjugacy (Legendre-Fenchel transform), which says that any μ from the interior of $\operatorname{conv}\phi(X)$ and any θ satisfy Fenchel's inequality

$$F(\theta) - H(\mu) - \theta\mu \geq 0 \tag{5}$$

where equality holds if and only if $\mu = m(\theta)$. An alternative view of (5) is that $F(\theta) - H(\mu) - \theta\mu$ is the KL-divergence between the distribution determined by θ and the distribution determined by μ.

Notice that the equality (4) can be obtained by minimizing the left-hand side of (5) with respect to θ. Similarly, minimizing with respect to μ yields

$$\frac{\mathrm{d}H(\mu)}{\mathrm{d}\mu} = -\theta \tag{6}$$

where θ is the (unique) solution of $m(\theta) = \mu$.

2.3 Reparameterizations

Now, suppose the basis functions ϕ_i are affinely dependent, that is, they form an overcomplete representation of the family. These dependencies can be written as

$$A\phi(x) = 0,\quad B\phi(x) = 1 \qquad \forall x \in X \tag{7}$$

[1] These saddle points should not be confused with the saddle points in the double-loop algorithms to minimize the Bethe free energy (Heskes, 2003; Heskes, 2006).

for some matrices A and B, where 0 and 1 denote here column vectors of zeros and ones. Thus, matrix A captures homogeneous dependencies and matrix B captures inhomogeneous dependencies. It follows that

$$\operatorname{aff}\phi(X) = \{\,\mu \in \mathbb{R}^I \mid A\mu = 0,\ B\mu = 1\,\}$$

is the affine hull of the set $\phi(X)$.

Let α and β be arbitrary row vectors and let

$$\theta' = \theta + \alpha A + \beta B \tag{8}$$

Then, $\theta'\phi(x) = \theta\phi(x) + \beta 1$ and $F(\theta') = F(\theta) + \beta 1$. It follows that transformation (8) preserves distribution (1) and it is thus a *reparameterization* of the distribution. We will refer to the subclass of reparameterizations with $\beta = 0$ as *homogeneous reparameterizations*.

For an overcomplete representation, θ is no longer determined by μ uniquely but only up to reparameterizations, $m(\theta + \alpha A + \beta B) = m(\theta)$.

Moreover, equality (6) can no longer be used because the partial derivatives of $H(\mu)$ are undefined – only directional derivatives parallel to the space $\operatorname{aff}\phi(X)$ are defined. Let

$$\nabla_\nu H(\mu) = \lim_{t\to 0} \frac{H(\mu + t\nu) - H(\mu)}{t} = \left.\frac{\mathrm{d}H(\mu + t\nu)}{\mathrm{d}t}\right|_{t=0}$$

denote the directional derivative of $H(\mu)$ in direction $\nu \in \mathbb{R}^I$. To be parallel to $\operatorname{aff}\phi(X)$, ν has to satisfy $A\nu = 0$ and $B\nu = 0$. Then (6) generalizes to

$$\nabla_\nu H(\mu) = -\theta\nu \qquad \forall\nu\colon A\nu = 0,\ B\nu = 0 \tag{9}$$

This is consistent with the fact that $\theta\nu$ is invariant to reparameterizations, $(\theta + \alpha A + \beta B)\nu = \theta\nu$.

3 Gibbs distribution

In §3, we show how the Gibbs distribution on a graphical model arises as a special exponential family.

Let V be a set of variables. Let $E \subseteq 2^V$ be a set variable subsets, i.e., (V, E) is a hypergraph[2]. We assume E contains no one-element subsets. A variable v takes states $x_v \in X_v$, where X_v is a finite domain of the variable. For a hyperedge $a \in E$, let $X_a = \bigtimes_{v\in a} X_v$ denote the Cartesian product of domains of variables a. Elements of X_a will be denoted x_a.

We instantiate (X, I, ϕ) such that distribution (1) becomes the Gibbs distribution on hypergraph (V, E). Let $X = X_V$ be the Cartesian product of all variable domains. Let

$$I = \{(v, x_v) \mid v \in V,\ x_v \in X_v\} \cup \{(a, x_a) \mid a \in E,\ x_a \in X_a\}$$

[2]Though we consider the factor-graph BP in the paper, we do not use the concept of a factor graph – we use a hypergraph instead, which is clearly equivalent.

For $i \in I$, we denote the i-component of vector θ and μ by $\theta_v(x_v), \theta_a(x_a)$ and $\mu_v(x_v), \mu_a(x_a)$, respectively. Let $\phi\colon X \to \{0,1\}^I$ be indicator functions chosen such that

$$\theta\phi(x) = \sum_{v\in V} \theta_v(x_v) + \sum_{a\in E} \theta_a(x_a) \tag{10}$$

Now, distribution (1) is the Gibbs distribution and $\mu = m(\theta)$ are its marginals,

$$\mu_v(x_v) = \sum_{x_{V\setminus v}} p(x\,|\,\theta), \qquad \mu_a(x_a) = \sum_{x_{V\setminus a}} p(x\,|\,\theta)$$

The polytope $\operatorname{conv}\phi(X)$ contains all realizable marginal vectors μ and is known as the *marginal polytope* (Wainwright & Jordan, 2008). Moreover, for this choice of (X, I, ϕ) we have $\{0,1\}^I \cap \operatorname{aff}\phi(X) = \phi(X)$.

3.1 Affine dependencies

Now we specify matrices A and B, which capture affine dependencies among functions ϕ_i. We do this indirectly, by writing down products $A\mu$, $B\mu$, αA and βB.

Equation systems $A\mu = 0$ and $B\mu = 1$ turn out to be the familiar marginalization and normalization conditions, respectively:

$$\sum_{x_{a\setminus v}} \mu_a(x_a) - \mu_v(x_v) = 0 \tag{11a}$$

$$\sum_{x_v} \mu_v(x_v) = 1, \quad \sum_{x_a} \mu_a(x_a) = 1 \tag{11b}$$

Let us remark that (11a) describes only a subset of all existing homogeneous dependencies among ϕ_i, namely those that couple hyperedges with variables, and omits those that couple pairs of hyperedges. But this is a limitation of the factor-graph BP compared to the generalized BP. All existing dependences would be described by $\sum_{x_{a\setminus b}} \mu_a(x_a) = \sum_{x_{b\setminus a}} \mu_b(x_b)$. If $E \subseteq \binom{V}{2}$ is an ordinary graph (that means, there are only pairwise interactions), (11a) describes all existing dependencies.

Reparameterization $\theta' = \theta + \alpha A + \beta B$ reads

$$\theta'_v(x_v) = \theta_v(x_v) - \sum_{a\ni v} \alpha_{av}(x_v) + \beta_v \tag{12a}$$

$$\theta'_a(x_a) = \theta_a(x_a) + \sum_{v\in a} \alpha_{av}(x_v) + \beta_a \tag{12b}$$

Let us explain the detailed meaning of (12).

We define the *elementary homogeneous reparameterization* as follows: pick any pair (a, v) with $v \in a$, subtract an arbitrary unary function $\alpha_{av}(\cdot)$ from function $\theta_v(\cdot)$, and add the same function to $\theta_a(\cdot)$:

$$\theta'_v(x_v) \leftarrow \theta_v(x_v) - \alpha_{av}(x_v) \tag{13a}$$

$$\theta'_a(x_a) \leftarrow \theta_a(x_a) + \alpha_{av}(x_v) \tag{13b}$$

Since $\alpha_{av}(x_v)$ cancels out, this preserves the sum $\theta_v(x_v) + \theta_a(x_a)$ and hence also the function (10). Applying transformations (13) to all pairs (a, v) yields the terms with α in (12), i.e., the homogeneous reparameterization $\theta' = \theta + \alpha A$.

Reparameterization $\theta' = \theta + \beta B$ simply adds constants β_v, β_a to all functions $\theta_v(\cdot)$, $\theta_a(\cdot)$.

Let us point out that papers on graphical models usually mean by 'reparameterizations' only homogeneous reparameterizations, or are not explicit about that.

Reparameterizations in the form (12) and (13) were first used by (Shlezinger, 1976) in LP relaxation of the problem $\max_{x \in X} \theta\phi(x)$ (i.e., finding modes of a Gibbs distribution). More can be found in modern revisions (Werner, 2007; Werner, 2010) of this approach.

4 Belief propagation

In the most general formulation (Yedidia et al., 2005), BP and related algorithms and free energies start with decomposing the original hypergraph into a collection of sub-hypergraphs (typically, hypertrees). Each sub-hypergraph is assigned a *counting number* (negative, zero, or positive) such that every hyperedge of the original hypergraph is counted exactly once in total.

In the factor-graph BP, our hypergraph (V, E) is decomposed into the collection of sub-hypergraphs E^v and E^a, where $v \in V$ and $a \in E$. Hypergraph E^v contains only variable v. Hypergraph E^a contains hyperedge a and variables $v \in a$. The counting number of E^a equals 1 and the counting number of E^v equals $1 - n_v$, where $n_v = \sum_{a \ni v} 1$.

Each sub-hypergraph defines its own local Gibbs distribution. Let the distribution on E^v and E^a be denoted respectively by

$$p^v(x_v \,|\, \theta) = \exp\left[\theta_v(x_v) - F^v(\theta)\right] \tag{14a}$$

$$p^a(x_a \,|\, \theta) = \exp\left[\theta_a(x_a) + \sum_{v \in a} \theta_v(x_v) - F^a(\theta)\right] \tag{14b}$$

where the local log-partition functions read

$$F^v(\theta) = \bigoplus_{x_v} \theta_v(x_v) \tag{15a}$$

$$F^a(\theta) = \bigoplus_{x_a} \left[\theta_a(x_a) + \sum_{v \in a} \theta_v(x_v)\right] \tag{15b}$$

Similarly, the entropies of distributions (14) read[3]

$$H^v(\mu) = -\sum_{x_v} \mu_v(x_v) \log \mu_v(x_v) \tag{16a}$$

$$H^a(\mu) = -\sum_{x_a} \mu_a(x_a) \log \mu_a(x_a) \tag{16b}$$

[3]It might seem surprising that numbers $\mu_v(x_v)$ for $v \in a$ are absent in (16b). But (16b) is correct, variables really have zero counting numbers in hypergraph E^a.

Let us define two functions

$$\tilde{F}(\theta) = \sum_{v \in V} (1 - n_v) F^v(\theta) + \sum_{a \in E} F^a(\theta) \tag{17}$$

$$\tilde{H}(\mu) = \sum_{v \in V} (1 - n_v) H^v(\mu) + \sum_{a \in E} H^a(\mu) \tag{18}$$

While the function $\tilde{H}$ is the well-known Bethe entropy approximation, $\tilde{F}$ can be seen as the 'Bethe log-partition function'. To our knowledge, the function $\tilde{F}$ was not mentioned in previous works.

Next we proceed as follows. In §4.1 we define the BP algorithm and its fixed points. Then we give two interpretations of BP fixed points:

- In §4.2 we recall the well-known result by (Yedidia et al., 2000; Yedidia et al., 2005) that BP fixed points correspond to stationary points of the (negative) Bethe free energy $\theta\mu + \tilde{H}(\mu)$ on the space $\{\, \mu > 0 \mid A\mu = 0,\ B\mu = 1 \,\}$. We refer to this as the *dual interpretation*.
- In §4.3 we present our main result, that BP fixed points correspond to stationary points of the function $\tilde{F}(\theta)$ on the space of homogeneous reparameterizations of θ. We refer to this as the *primal interpretation*.

Here, we use the term 'stationary point' in a slightly broader meaning than is usual: a *stationary point of a function on an affine space* is a point where all directional derivatives parallel to that space vanish.

4.1 BP algorithm and its fixed points

Usually, BP is formulated in terms of passing messages, following (Pearl, 1988). We formulate it here in terms of reparameterizations. Our formulation is related to but different from (Wainwright et al., 2004).

In BP, probabilities (14) are seen as approximations of the true variable and hyperedge marginals of the Gibbs distribution (1). For a general θ, they fail to satisfy the marginal consistency condition

$$\sum_{x_{a \setminus v}} p^a(x_a \,|\, \theta) = p^v(x_v \,|\, \theta) \tag{19}$$

which has to be satisfied by true marginals. The BP algorithm seeks to reparameterize θ such that (19) holds. Since functions (14) are invariant to reparameterizations $\theta' = \theta + \beta B$, only homogeneous reparameterizations can be considered. Plugging (14) into (19) yields

$$\bigoplus_{x_{a \setminus v}} \left[\theta_a(x_a) + \sum_{u \in a \setminus v} \theta_u(x_u)\right] = \mathrm{const}_{av} \tag{20}$$

where $\mathrm{const}_{av} = F^a(\theta) - F^v(\theta)$ are constants independent on x_v. We define a *BP fixed point* to be a vector θ satisfying (20).

A single update of the BP algorithm (its serial version) enforces condition (20) to hold for a single pair (a, v) by applying the elementary homogeneous reparameterization (13) to the pair (a, v). This determines $\alpha_{av}(\cdot)$ in (13) up to a constant. This constant is set so that $\bigoplus_{x_v} \alpha_{av}(x_v) = \bigoplus_{x_v} 0$, which ensures that numbers θ stay bounded during the algorithm.

In our exponential family formalism, the BP fixed point condition can be stated concisely as follows. Let a map $\mu = \tilde{m}(\theta)$ be defined by $\mu_v(x_v) = p^v(x_v\,|\,\theta)$, $\mu_a(x_a) = p^a(x_a\,|\,\theta)$. Map $\tilde{m}$ can be seen as an approximation of the true marginal map m. Now, BP fixed point condition (19) reads simply $A\tilde{m}(\theta) = 0$.

The true map m satisfies $Am(\theta) = 0$, $Bm(\theta) = 1$ and $m(\theta + \alpha A + \beta B) = m(\theta)$. In contrast, $\tilde{m}$ satisfies only $B\tilde{m}(\theta) = 1$ and $\tilde{m}(\theta + \beta B) = \tilde{m}(\theta)$ in general. BP seeks to reparameterize θ such that also $A\tilde{m}(\theta) = 0$, i.e., to solve the system $A\tilde{m}(\theta + \alpha A) = 0$ for α.

4.2 Dual interpretation of BP

In variational inference (Wainwright & Jordan, 2008), the log-partition function F and marginals m are computed indirectly via convex conjugacy between F and $-H$. Fenchel's inequality (5) implies that

$$F(\theta) = \max\{\, \theta\mu + H(\mu) \mid \mu > 0,\, \mu \in \operatorname{conv}\phi(X) \,\} \quad (21)$$

where the optimum is attained at $\mu = m(\theta)$. This so far provides no advantage because both the marginal polytope $\operatorname{conv}\phi(X)$ and the entropy function H are defined in an intractable way. The trick is to replace them with their tractable approximations. Then, the optimal argument and value of (21) is an approximation of the true $m(\theta)$ and $F(\theta)$, respectively.

If the polytope $\operatorname{conv}\phi(X)$ is approximated with the 'local polytope' (Wainwright & Jordan, 2008)

$$[0,1]^I \cap \operatorname{aff}\phi(X) = \{\, \mu \geq 0 \mid A\mu = 0,\; B\mu = 1 \,\} \quad (22)$$

and the true entropy H with the Bethe entropy (18), we obtain the Bethe variational problem

$$\max\{\, \theta\mu + \tilde{H}(\mu) \mid \mu > 0,\; A\mu = 0,\; B\mu = 1 \,\} \quad (23)$$

where $-\theta\mu - \tilde{H}(\mu)$ is known as the Bethe free energy.

In general, $[0,1]^I \cap \operatorname{aff}\phi(X) \supset \operatorname{conv}\phi(X)$ and $\tilde{H} \neq H$. However, if the factor graph of our graphical model is acyclic then $[0,1]^I \cap \operatorname{aff}\phi(X) = \operatorname{conv}\phi(X)$ and $\tilde{H} = H$. (Wainwright & Jordan, 2008; Yedidia et al., 2005).

Let us emphasize that the BP algorithm does not directly solve problem (23). BP maintains $\mu = \tilde{m}(\theta)$, which ensures $\mu > 0$ and $B\mu = 1$, and tries to reparameterize θ so that $A\mu = 0$. Thus, μ is infeasible to (23) until BP converges. Operating on the Lagrange multipliers of (23), BP is a *dual* algorithm to solve (23).

(Yedidia et al., 2005) showed that BP fixed points correspond to stationary points of problem (23). We need to say precisely what is meant by this correspondence because we defined BP fixed points in terms of θ and stationary points of (23) in terms of μ. The correspondence is given by the map $\mu = \tilde{m}(\theta)$. This map is one-to-one up to adding constants to functions $\theta_v(\cdot), \theta_a(\cdot)$, i.e., up to reparameterizations $\theta \leftarrow \theta + \beta B$.

Moreover, notice that the objective of (23) is invariant to homogeneous reparameterizations because $(\theta + \alpha A)\mu = \theta\mu$ for feasible μ.

With this understanding, we can state Yedidia's result.

Theorem 1. *If θ and μ correspond through $\mu = \tilde{m}(\theta)$, the following statements are equivalent:*

- *$A\tilde{m}(\theta) = 0$, i.e., θ is a BP fixed point.*
- *μ is a stationary point of $\theta\mu + \tilde{H}(\mu)$ on the set $\{\, \mu > 0 \mid A\mu = 0,\; B\mu = 1 \,\}$.*

Let us remark that, by the discussion in §2.3, the second statement says that the directional derivative of $\theta\mu + \tilde{H}(\mu)$ vanishes in all directions parallel to $\operatorname{aff}\phi(X)$:

$$\nabla_\nu \tilde{H}(\mu) = -\theta\nu \qquad \forall \nu\colon A\nu = 0,\; B\nu = 0 \quad (24)$$

4.3 Primal interpretation of BP

Here we present our main result, which can be concisely stated as follows: the BP algorithm tries to find a vector α such that the gradient of $\tilde{F}(\theta + \alpha A)$ with respect to α vanishes.

This gradient can be conveniently evaluated at $\alpha = 0$ without loss of generality since the gradient at $\alpha \neq 0$ can be recovered by replacing θ with $\theta + \alpha A$. Thus, we claim that θ is a BP fixed point if and only if

$$\left.\frac{\mathrm{d}\tilde{F}(\theta + \alpha A)}{\mathrm{d}\alpha}\right|_{\alpha=0} = A\frac{\mathrm{d}\tilde{F}(\theta)}{\mathrm{d}\theta} = 0 \quad (25)$$

where the first equality follows from the chain rule.

An alternative interpretation of condition (25) is that θ is a stationary point of function $\tilde{F}(\theta)$ on the space of homogeneous reparameterizations of θ. Recall that this is the space of vectors $\theta + \alpha A$ for all possible α. Condition (25) says that all directional derivatives parallel to this space vanish. Now we formulate our result.

Theorem 2. *The following statements are equivalent:*

- *$A\tilde{m}(\theta) = 0$, i.e., θ is a BP fixed point.*
- *$A\,[\mathrm{d}\tilde{F}(\theta)/\mathrm{d}\theta] = 0$, i.e., θ is a stationary point of $\tilde{F}(\theta)$ on the space of homogeneous reparameterizations of θ.*

Proof. In the first part of the proof, we express the derivative (25) in terms of $\tilde{m}(\theta)$.

We begin by expressing the derivative $\mathrm{d}\tilde{F}(\theta)/\mathrm{d}\theta$ in terms of $\tilde{m}(\theta)$. Differentiating (17) yields

$$\frac{\partial \tilde{F}(\theta)}{\partial \theta_v(x_v)} = (1-n_v)\frac{\partial F^v(\theta)}{\partial \theta_v(x_v)} + \sum_{a \ni v} \frac{\partial F^a(\theta)}{\partial \theta_v(x_v)} \quad (26a)$$

$$\frac{\partial \tilde{F}(\theta)}{\partial \theta_a(x_a)} = \frac{\partial F^a(\theta)}{\partial \theta_a(x_a)} \quad (26b)$$

Let us denote $\mu = \tilde{m}(\theta)$ for brevity. By (4), we have

$$\frac{\partial F^v(\theta)}{\partial \theta_v(x_v)} = \mu_v(x_v)$$
$$\frac{\partial F^a(\theta)}{\partial \theta_a(x_a)} = \mu_a(x_a) \qquad \frac{\partial F^a(\theta)}{\partial \theta_v(x_v)} = \sum_{x_{a \setminus v}} \mu_a(x_a)$$

Plugging this into (26) and some manipulations yields

$$\frac{\partial \tilde{F}(\theta)}{\partial \theta_v(x_v)} = \mu_v(x_v) + \sum_{a \ni v} \gamma_{av}(x_v) \quad (27a)$$

$$\frac{\partial \tilde{F}(\theta)}{\partial \theta_a(x_a)} = \mu_a(x_a) \quad (27b)$$

where we denoted $\gamma = A\mu$, i.e.,

$$\gamma_{av}(x_v) = \sum_{x_{a \setminus v}} \mu_a(x_a) - \mu_v(x_v)$$

By (11a), the components of (25) read

$$\left.\frac{\partial \tilde{F}(\theta + \alpha A)}{\partial \alpha_{av}(x_v)}\right|_{\alpha=0} = \sum_{x_{a \setminus v}} \frac{\partial \tilde{F}(\theta)}{\partial \theta_a(x_a)} - \frac{\partial \tilde{F}(\theta)}{\partial \theta_v(x_v)} \quad (28)$$

Plugging (27) into (28) finally yields

$$\left.\frac{\partial \tilde{F}(\theta + \alpha A)}{\partial \alpha_{av}(x_v)}\right|_{\alpha=0} = -\sum_{b \ni v,\, b \neq a} \gamma_{bv}(x_v) \quad (29)$$

In the second part of the proof, we express the two statements in Theorem 2 in terms of γ. The first statement is equivalent to system (30a) below. By (29), the second statement is equivalent to system (30b).

$$\gamma_{av}(x_v) = 0 \qquad \forall a \in E,\ v \in a,\ x_v \quad (30a)$$

$$\sum_{b \ni v,\, b \neq a} \gamma_{bv}(x_v) = 0 \qquad \forall a \in E,\ v \in a,\ x_v \quad (30b)$$

We need to show that systems (30a) and (30b) are equivalent. This can be shown separately for each pair (v, x_v). Pick (v, x_v) and write γ_a instead of $\gamma_{av}(x_v)$ for simplicity. Then we need to show that an arbitrary set of numbers $\{\, \gamma_a \mid a \ni v \,\}$ satisfies the equivalence

$$\Big[\, \gamma_a = 0 \ \ \forall a \ni v \,\Big] \iff \Big[\, \sum_{b \ni v,\, b \neq a} \gamma_b = 0 \ \ \forall a \ni v \,\Big]$$

which is already easy. ∎

Next, we give a second order property of function $\tilde{F}$.

Theorem 3. *Consider $\tilde{F}(\theta + \alpha A)$ as a function of α. Every stationary point of this function is a saddle point.*

Proof. We need to show that the Hessian

$$\frac{\mathrm{d}^2 \tilde{F}(\theta + \alpha A)}{\mathrm{d}\alpha^2}$$

is indefinite at any point α satisfying $A\tilde{m}(\theta + \alpha A) = 0$. It suffices to show that only a partial Hessian is indefinite. We obtain this partial Hessian by computing the partial derivatives $\partial^2 \tilde{F}(\theta + \alpha A)/\partial \alpha_k\, \partial \alpha_\ell$ only for some of all possible pairs (k, ℓ). After some work (we do not present details of the derivation) we get

$$\frac{\partial^2 \tilde{F}(\theta + \alpha A)}{\partial \alpha_{av}(x_v)\, \partial \alpha_{bv}(x_v)} = \begin{cases} 0 & \text{if } a = b \\ [\mu_v(x_v) - 1]\mu_v(x_v) & \text{if } a \neq b \end{cases}$$

where $\mu = \tilde{m}(\theta + \alpha A)$. This holds only if $A\mu = 0$, at points $A\mu \neq 0$ the derivative is more complex. The derivative takes only two values, depending on whether $a = b$ or $a \neq b$. Hence the diagonal elements of the partial Hessian are zero and the remaining elements are equal. Any such matrix is indefinite. ∎

4.4 Relation of primal and dual view

One can notice that $\tilde{F}$, $\tilde{H}$, $\tilde{m}$ are related by certain equalities, which can be seen as 'rudiments' of convex conjugacy relationship among the true F, H, m.

Thus, (27) shows that if θ is a BP fixed point then

$$\frac{\mathrm{d}\tilde{F}(\theta)}{\mathrm{d}\theta} = \tilde{m}(\theta) \quad (31)$$

In contrast to (4), equality (31) holds only at BP fixed points because of the extra term $\sum_{a \ni v} \gamma_{av}(x_v)$ in (27a), which vanishes at and only at BP fixed points. In fact, this might suggest that the map $\mu = \mathrm{d}\tilde{F}(\theta)/\mathrm{d}\theta$ is a more fundamental object than the map $\mu = \tilde{m}(\theta)$ – but we do not further pursue this observation here.

If $\mu = \tilde{m}(\theta)$ and $A\mu = 0$ (i.e., θ is a BP fixed point) then

$$\tilde{F}(\theta) - \tilde{H}(\mu) - \theta\mu = 0 \quad (32)$$

Unlike Fenchel's equality for true F, H, m, equality (32) fails to hold if $A\mu \neq 0$. Interestingly, we observed that condition $A\mu = 0$ becomes unnecessary if the form (18) of the Bethe entropy is replaced by its different form. Let

$$\tilde{H}(\mu) = \sum_{v \in V} H^v(\mu) - \sum_{a \in E} J^a(\mu) \quad (33)$$

where

$$J^a(\mu) = \sum_{x_a} \mu_a(x_a) \log \frac{\mu_a(x_a)}{\prod_{v \in a} \mu_v(x_v)}$$

is the KL-divergence between $\mu_a(x_a)$ and $\prod_{v\in a}\mu_v(x_v)$. Functions (18) and (33) are equal for $A\mu = 0$ but different otherwise (Wainwright & Jordan, 2008, §4.1.2). It can be easily verified that with this form of $\tilde{H}$, equality (32) holds for $\mu = \tilde{m}(\theta)$ even if $A\mu \neq 0$. In other words, substitution $\mu = \tilde{m}(\theta)$ transforms the function $\theta\mu + \tilde{H}(\mu)$ into $\tilde{F}(\theta)$.

The Bethe entropy has a clear meaning: for acyclic graphs, $\tilde{H}$ equals the true entropy H. It follows from (32) that $\tilde{F}$ has a similar property: for acyclic graphs, $\tilde{F}$ equals the true log-partition function F but only if $A\tilde{m}(\theta) = 0$ (i.e., only on the space of BP fixed points).

5 Conclusion

We have presented a novel interpretation of loopy belief propagation. While it was known that BP fixed points correspond to stationary points of the Bethe free energy on the local polytope, we have shown that they also correspond to stationary points of the 'Bethe log-partition function' on the space of homogeneous reparameterizations. To the best of our knowledge, this simple observation was not made before. The two interpretations are exactly complementary – however, they are not related by classical convex duality because function $\tilde{H}$ is not concave and $\tilde{F}$ is not convex.

So far, BP was understood as an algorithm to seek for a common zero of a set of explicitly unrelated equations. Our result shows that these equations are partial derivatives of the single function $\tilde{F}(\theta + \alpha A)$ of α without any additional constraints.

One would expect that finding a stationary point of a single multivariate analytic function must be easier than solving a system of unrelated non-linear equations – but this is true only if the stationary point is a local extreme. Unfortunately, all stationary points of the function $\tilde{F}(\theta+\alpha A)$ are saddle points, and finding a saddle point can be much harder (and little literature seems to exist about it). Therefore, we currently do not know whether our result can provide new insights into (non-)convergence of BP.

Various generalized versions of BP are often designed via dual considerations involving local free energies and entropies. Our result suggests that free energies may not be needed for this at all, the primal route via reparameterizations and local log-partition functions may be simpler. This is open to future research.

Although we have not demonstrated any practical consequences of our contribution, we believe that the presented mathematical framework, which treats reparameterizations explicitly and incorporates them into the exponential family language, brings more clarity in the theoretical understanding of graphical models.

Acknowledgements

This research was supported by the European Commission grant 215078 (DIPLECS) and the Czech government grant MSM6840770038.

References

Heskes, T. (2003). Stable fixed points of loopy belief propagation are minima of the Bethe free energy. *Advances in Neural Information Processing Systems (NIPS)* (pp. 359–366).

Heskes, T. (2006). Convexity arguments for efficient minimization of the Bethe and Kikuchi free energies. *Jr. of Artificial Intelligence Research*, *26*, 153–190.

Kschischang, F. R., Frey, B. J., & Loeliger, H. A. (2001). Factor graphs and the sum-product algorithm. *IEEE Trans. Inf. Theory*, *47*, 498–519.

Pearl, J. (1988). *Probabilistic reasoning in intelligent systems: Networks of plausible inference*. San Francisco: Morgan Kaufmann.

Shlezinger, M. I. (1976). Syntactic analysis of two-dimensional visual signals in noisy conditions. *Cybernetics and Systems Analysis*, *12*, 612–628. Translation from Russian.

Wainwright, M., Jaakkola, T., & Willsky, A. (2004). Tree consistency and bounds on the performance of the max-product algorithm and its generalizations. *Statistics and Computing*, *14*, 143–166.

Wainwright, M. J., & Jordan, M. I. (2008). Graphical models, exponential families, and variational inference. *Foundations and Trends in Machine Learning*, *1*, 1–305.

Werner, T. (2007). A linear programming approach to max-sum problem: A review. *IEEE Trans. Pattern Analysis and Machine Intelligence*, *29*, 1165–1179.

Werner, T. (2010). Revisiting the linear programming relaxation approach to Gibbs energy minimization and weighted constraint satisfaction. *IEEE Trans. Pattern Analysis and Machine Intelligence*. To appear in August 2010.

Yedidia, J., Freeman, W. T., & Weiss, Y. (2000). Generalized belief propagation. *Neural Information Processing Systems (NIPS)* (pp. 689–695).

Yedidia, J. S., Freeman, W. T., & Weiss, Y. (2005). Constructing free-energy approximations and generalized belief propagation algorithms. *IEEE Trans. Information Theory*, *51*, 2282–2312.