# 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 $\psi$ 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

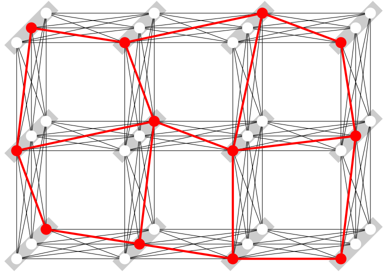
This section defines a class of labeling problems of which the CSP and the max-sum problem are special cases. Here we introduce basic terminology used in the rest of the article.

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



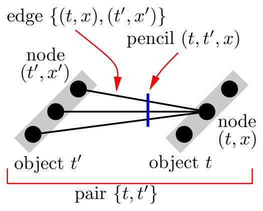


Fig. 1. (a) The $3 \times 4$ grid graph $G$ (i.e., $|T| = 12$), graph $(T \times X, E_X)$ for $|X| = 3$ labels, and a labeling $\mathbf{x}$ (emphasized). (b) Parts of both graphs.

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

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

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.

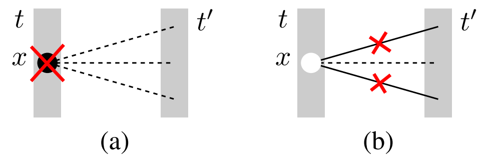


Fig. 2. The arc consistency algorithm deletes (a) nodes not linked with some neighbor by any edge, and (b) edges lacking an end node.

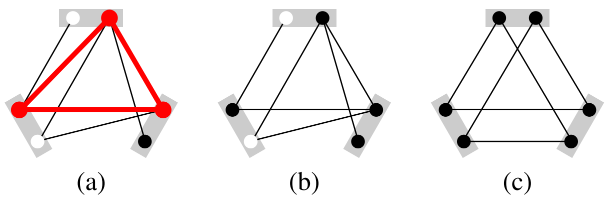


Fig. 3. Examples of CSPs: (a) satisfiable, hence with a non-empty kernel which allows to form a labeling (the labeling is emphasized); (b) with an empty kernel, hence unsatisfiable; (c) arc consistent but unsatisfiable. The forbidden nodes are in white, the forbidden edges are not shown.

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 $\alpha_t$ 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 $\mathbf{g}^{\boldsymbol{\varphi}} = \mathbf{g} + \mathbf{0}^{\boldsymbol{\varphi}}$.

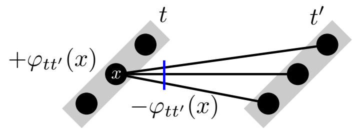


Fig. 4. The elementary equivalent transformation.

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

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

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

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 $\varphi_t$ 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 $\varphi_{tt'}(x)$ += $\varphi_t$ for all $x$ and $\varphi_{t't}(x')$ −= $\varphi_t$ for all $x'$. Set $\varphi_{t'}$ += $\varphi_t$ and $\varphi_t := 0$. 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 $\boldsymbol{\varphi} \in \mathbb{R}^P$.

$$
\begin{array}{rcll}
\langle \mathbf{g}, \boldsymbol{\alpha} \rangle \to \max\limits_{\boldsymbol{\alpha}} & \displaystyle\sum_{t\in T} u_t + \sum_{\{t,t'\}\in E} u_{tt'} \to \min\limits_{\boldsymbol{\varphi},\mathbf{u}} & & \text{(11a)}\\
\displaystyle\sum_{x'\in X} \alpha_{tt'}(x,x') = \alpha_t(x) & \varphi_{tt'}(x) \in \mathbb{R}, & \{t,t'\} \in E,\ x \in X & \text{(11b)}\\
\displaystyle\sum_{x\in X} \alpha_t(x) = 1 & u_t \in \mathbb{R}, & t \in T & \text{(11c)}\\
\displaystyle\sum_{x,x'\in X} \alpha_{tt'}(x,x') = 1 & u_{tt'} \in \mathbb{R}, & \{t,t'\} \in E & \text{(11d)}\\
\alpha_t(x) \geq 0 & u_t - \displaystyle\sum_{t'\in N_t} \varphi_{tt'}(x) \geq g_t(x), & t \in T,\ x \in X & \text{(11e)}\\
\alpha_{tt'}(x,x') \geq 0 & u_{tt'} + \varphi_{tt'}(x) + \varphi_{t't}(x') \geq g_{tt'}(x,x'), & \{t,t'\} \in E,\ x,x' \in X & \text{(11f)}
\end{array}
$$

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

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

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

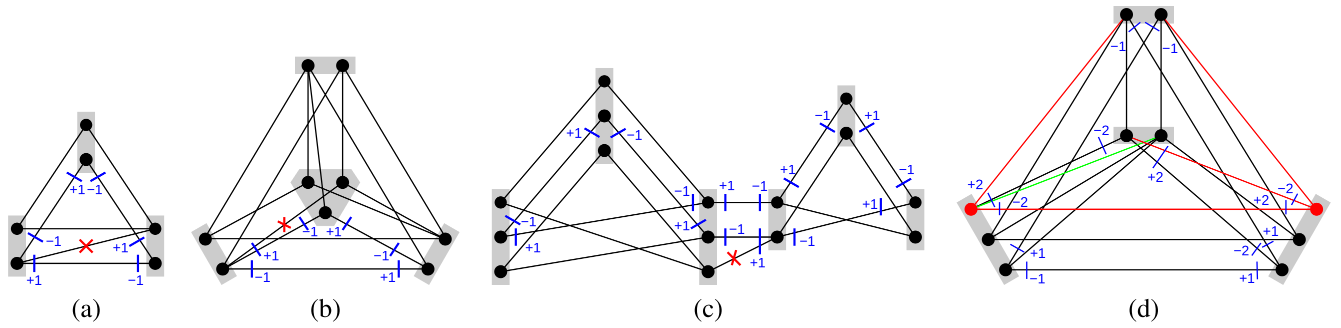


Fig. 5. Examples of kernels not invariant to equivalent transformations. The shown edges have quality 0 and the not shown edges $-\infty$. Problem (a) has minimal height, problems (b, c) do not; in particular, for (b, c) system (7a)+(7b)+(17) is unsolvable. For problem (d), system (7a)+(7b)+(17) is solvable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

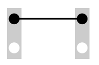
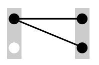
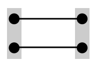
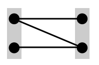
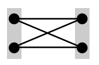
### C. Boolean Max-sum Problems

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 $(n_t, n_{t'}, n_{tt'})$:

| $(1,1,1)$ | $(1,2,2)$ | $(2,2,2)$ | $(2,2,3)$ | $(2,2,4)$ |
|---|---|---|---|---|
|  |  |  |  |  |

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.

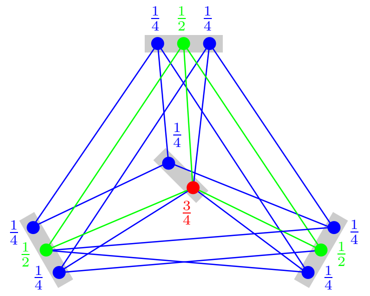


Fig. 6. A CSP for which $\bar{\Lambda}_{G,X}(\bar{\mathbf{g}})$ has a single element $\boldsymbol{\alpha}$ that is not an integer multiple of $|X|^{-1}$. This can be verified by solving system (7a)+(7b)+(17).

### D. Summary: Three Kinds of Consistency

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

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

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
    $\varphi_{tt'}(x) \mathrel{+}= \frac{1}{2}[\max_{x'} g^{\varphi}_{tt'}(x, x') - g^{\varphi}_t(x)]$;
  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 $\max_x f_1(x) + \max_x f_2(x) \geq \max_x [f_1(x) + f_2(x)]$. ∎

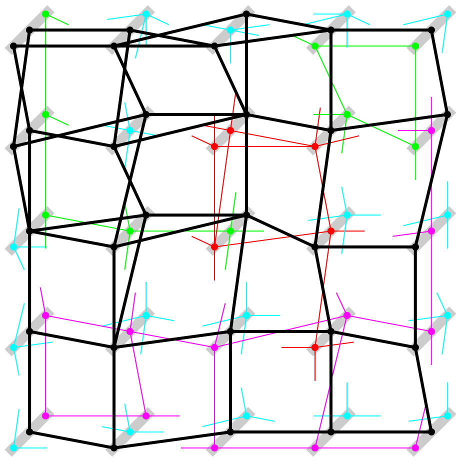


Fig. 7. A max-sum problem satisfying (18). A line segment starting from node $(t, x)$ and aiming to but not reaching $(t', x')$ denotes an edge satisfying $g_t(x) = g_{tt'}(x, x') < g_{t'}(x')$. If $g_t(x) = g_{tt'}(x, x') = g_{t'}(x')$, the line segment joins the nodes $(t, x)$ and $(t', x')$. The grey 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, $\varepsilon$. 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, $g_{tt'}^{\varphi}(x,x') \le 0$. 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 $T = \{a, \ldots, f\}$ 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. ■

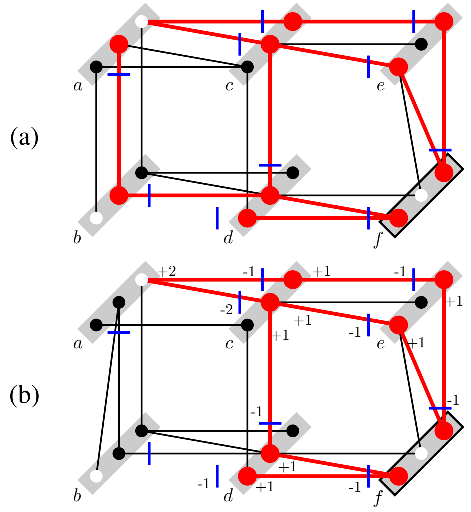


Fig. 8. (a) The augmenting DAG algorithm after phase 1; (b) after phase 2.

### 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 $\lambda$.

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 $\lambda$ 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 $\lambda$ 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 $\lambda$ 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 $\lambda$ 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 $\varepsilon$ is reasonably large, 'nearly maximal' nodes and edges are considered maximal and often a larger $\lambda$ 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 $\varepsilon$.

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 $U(\mathbf{g})$.

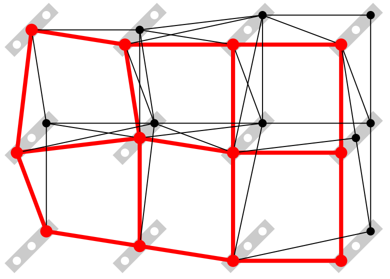
## VIII. Supermodular Max-sum Problems

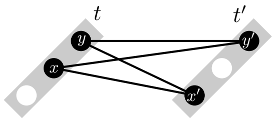
(Super-) submodularity, for bivariate functions also known as *(inverse) Monge property* [62], is well-known to simplify many optimization tasks; in fact, it can be considered a discrete counterpart of convexity [63]. It is long known that set supermodular max-sum problems can be translated to max-flow/min-cut [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


(a)



(b)

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.

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.

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

$$f(a \wedge b) + f(a \vee b) \ge f(a) + f(b). \qquad (20)$$

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 $F(\bullet \mid \mathbf{g})$ 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 $F(\bullet \mid \mathbf{g})$ is supermodular, by (20) it is $F(\mathbf{x} \wedge \mathbf{y} \mid \mathbf{g}) + F(\mathbf{x} \vee \mathbf{y} \mid \mathbf{g}) \geq F(\mathbf{x} \mid \mathbf{g}) + F(\mathbf{y} \mid \mathbf{g})$. 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 $F(\bullet \mid \mathbf{g})$ 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 $F(\bullet \mid \mathbf{g})$. This completes the proof that every supermodular max-sum problem has a trivial equivalent and is tractable.

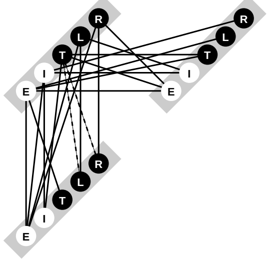
## IX. Application to Structural Image Analysis

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.

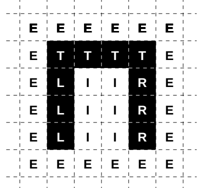
Let the following be given. Let $G$ represent a 4-connected image grid. Each pixel $t \in T$ has a label from $X = \{\mathsf{E}, \mathsf{I}, \mathsf{T}, \mathsf{L}, \mathsf{R}\}$. Numbers $g_{tt'}(x, x')$ are given by figure 10a, which shows three pixels forming one horizontal and one vertical pair, 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 $f(\mathsf{E}) = f(\mathsf{I}) = 1$ and $f(\mathsf{T}) = f(\mathsf{L}) = f(\mathsf{R}) = 0$ 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 $\{\, \mathbf{f}(\mathbf{x}) \mid F(\mathbf{x} \mid \mathbf{g}) > -\infty \,\}$ contains images feasible to the 2D grammar $(G, X, \mathbf{g})$, here, images of multiple non-overlapping black 'free-form' characters '$\Pi$' on white background. An example of such an image with labels denoted is in figure 10b. The number of characters in the image is $-F(\mathbf{x} \mid \mathbf{g})$.



(a)



(b)

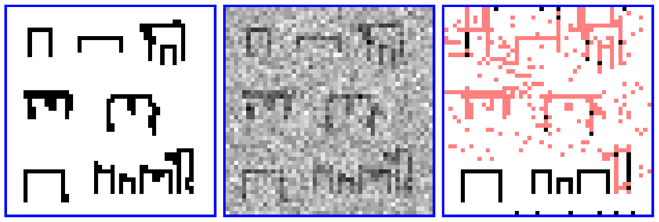

(c) (d) (e)

Fig. 10. The 'Letters $\Pi$' 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 $50 \times 50$ 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 $\{\, \mathbf{f}(\mathbf{x}) \mid F(\mathbf{x} \mid \mathbf{g}) > -\infty \,\}$, 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

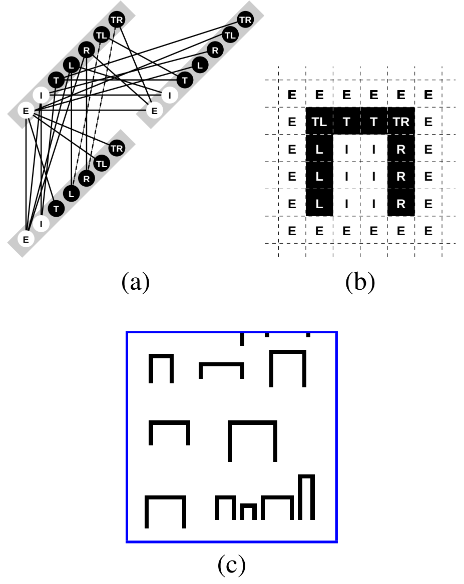


Fig. 11. The ‘Letters Π 2’ example, alternative ‘better’ definition of ’Letters Π’. (a) Definition, (b) a feasible labeled image, (c) output. The input was figure 10d.

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 $\alpha$-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.