# Universality of the Local Marginal Polytope

Daniel Průša and Tomáš Werner
Center for Machine Perception, Faculty of Electrical Engineering, Czech Technical University
Karlovo náměstí 13, 12135 Praha, Czech Republic
{prusapa1,werner}@cmp.felk.cvut.cz

## Abstract

*We show that solving the LP relaxation of the MAP inference problem in graphical models (also known as the min-sum problem, energy minimization, or weighted constraint satisfaction) is not easier than solving any LP. More precisely, any polytope is linear-time representable by a local marginal polytope and any LP can be reduced in linear time to a linear optimization (allowing infinite weights) over a local marginal polytope.*

## 1. Introduction

Given a set of discrete variables and a collection of functions each depending on one or two variables, the (pairwise) *min-sum problem* is defined as minimizing the sum of the functions over all the variables. This NP-complete combinatorial optimization problem occurs in MAP inference in graphical models [10], and is also known as *energy minimization* or *weighted constraint satisfaction* [3].

The problem has a natural LP relaxation, proposed independently in [9, 7, 2]. While the exact min-sum problem is equivalent to linear optimization over the *marginal polytope*, the LP relaxation approximates this polytope by its outer bound, known as the *local marginal polytope* [10].

We show that linear optimization over the local marginal polytope is in certain sense not easier than any linear program. In particular, we prove the following theorems.

**Theorem 1.** *Every polytope is (up to scale) a coordinate-erasing projection of a face of a local marginal polytope with 3 labels, whose description can be computed from the description of the original polytope in linear time.*

Here, by *polytope* we mean *bounded convex polyhedron*. A *coordinate-erasing projection* is a mapping $\pi\colon \mathbb{R}^q \to \mathbb{R}^p$ with $p < q$ given by $\pi(x_1, \dots, x_q) = (x_{\tau(1)}, \dots, x_{\tau(p)})$ for some injection $\tau\colon \{1, \dots, p\} \to \{1, \dots, q\}$.

**Theorem 2.** *Any linear program can be reduced in linear time to a linear optimization (allowing infinite weights) over a local marginal polytope with 3 labels.*

While Theorem 2 immediately follows from Theorem 1, the situation is more complex when infinite weights in the min-sum problem are not allowed. In this case, we show that the reduction can be done in quadratic time and space.

Similar universality result are known also for other polytopes, such as the three-way transportation polytope [4].

The most important consequence of our result is that it imposes a practical constraint on complexity of any algorithm to solve the LP relaxation of the min-sum problem. Designing a very efficient such algorithm might mean improving complexity (time complexity, or a combination of space and time complexity) of the best known algorithm for general linear programming, which is unlikely.

## 2. The local marginal polytope

Let $V$ be a finite set of *objects* and $E \subseteq \binom{V}{2}$ a set of *object pairs*, so that $(V, E)$ is an undirected graph. Let $K$ be a finite set of *labels*. The pairwise *min-sum problem* is defined as

$$\min_{\mathbf{k} \in K^V} \Big[ \sum_{u \in V} g_u(k_u) + \sum_{\{u,v\} \in E} g_{uv}(k_u, k_v) \Big] \tag{1}$$

where the functions $g_u\colon K \to \overline{\mathbb{R}}$ and $g_{uv}\colon K \times K \to \overline{\mathbb{R}}$ are unary and pairwise *interactions*, $\overline{\mathbb{R}} = \mathbb{R} \cup \{\infty\}$, and we adopt that $g_{uv}(k, \ell) = g_{vu}(\ell, k)$. We will refer to the values of $g_u$ and $g_{uv}$ as *weigths*. The values of all $g_u$ and $g_{uv}$ together will be understood as a vector $\mathbf{g} \in \overline{\mathbb{R}}^I$ with

$$\begin{aligned} I = &\{\, (u,k) \mid u \in V,\ k \in K \,\} \cup \\ &\{\, \{(u,k),(v,\ell)\} \mid \{u,v\} \in E;\ k,\ell \in K \,\}. \end{aligned} \tag{2}$$

The problem instance is defined by a tuple $(V, E, K, \mathbf{g})$.

Now we introduce the *local marginal polytope* [10]. It is the set $\Lambda$ of vectors $\boldsymbol{\mu} \in \mathbb{R}^I$ satisfying the constraints

$$\sum_{\ell \in K} \mu_{uv}(k, \ell) = \mu_u(k), \quad u \in V,\ v \in N_u,\ k \in K \tag{3a}$$

$$\sum_{k \in K} \mu_u(k) = 1, \qquad u \in V \tag{3b}$$

$$\boldsymbol{\mu} \geq \mathbf{0} \tag{3c}$$

where $N_u = \{ v \mid \{u,v\} \in E \}$ is the set of neighbors of object $u$ and we adopt $\mu_{uv}(k,\ell) = \mu_{vu}(\ell,k)$. The values of functions $\mu_u$ and $\mu_{uv}$ are known as *pseudomarginals*. The local marginal polytope is defined by a triplet $(V, E, K)$.

Now the LP relaxation of problem (1) reads

$$\Lambda^*(\mathbf{g}) = \operatorname*{argmin}_{\boldsymbol{\mu}\in\Lambda} \langle \mathbf{g}, \boldsymbol{\mu} \rangle, \tag{4}$$

where in the scalar product $\langle \mathbf{g}, \boldsymbol{\mu} \rangle$ we define $0 \cdot \infty = 0$. The set $\Lambda^*(\mathbf{g})$ contains all vectors $\boldsymbol{\mu}$ for which $\langle \mathbf{g}, \boldsymbol{\mu} \rangle$ attains its minimum over $\Lambda$. It is a polytope, namely a face of $\Lambda$. Conversely, every face of $\Lambda$ is $\Lambda^*(\mathbf{g})$ for some $\mathbf{g}$.

## 3. The input polyhedron

The input polyhedron is assumed to have the form

$$P = \{ \mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{R}^n \mid \mathbf{A}\mathbf{x} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0} \} \tag{5}$$

where $\mathbf{A} = [a_{ij}] \in \mathbb{Z}^{m\times n}$, $\mathbf{b} = (b_1, \ldots, b_m) \in \mathbb{Z}^m$, and $m \leq n$. Any convex polyhedron that can be represented by a finite number of bits can be described this way.

Before encoding, the system $\mathbf{A}\mathbf{x} = \mathbf{b}$ is rewritten as follows. In the $i$-th equation

$$a_{i1}x_1 + \cdots + a_{in}x_n = b_i \tag{6}$$

it is assumed that $b_i \geq 0$ (if not, multiply the whole equation by $-1$). Further, the terms with negative coefficients are moved to the other side of the equation, such that both sides have only non-negative terms. Precisely, (6) is rewritten as

$$a^+_{i1}x_1 + \cdots + a^+_{in}x_n = a^-_{i1}x_1 + \cdots + a^-_{in}x_n + b_i \tag{7}$$

where $a^+_{ij} \geq 0$, $a^-_{ij} \geq 0$, and $a_{ij} = a^+_{ij} - a^-_{ij}$.

Moreover, it is assumed that neither side of (7) vanishes for any $i$. If $a^-_{i1} = \cdots = a^-_{in} = b_i = 0$ and $a^+_{ij} > 0$ then inevitably $x_j = 0$ and thus $x_j$ can be eliminated from (5). If $a^+_{i1} = \cdots = a^+_{in} = 0$ and $b_i > 0$ then $P = \emptyset$.

Next we derive bounds that will be needed in the encoding algorithm. The following lemmas are not surprising but we state them with proofs for completeness of the paper.

**Lemma 3.** *For any matrix $\mathbf{A} \in \mathbb{R}^{n\times n}$ with columns $\mathbf{a}_j$,*

$$|\det \mathbf{A}| \leq n^{n/2} \prod_{j=1}^{n} \|\mathbf{a}_j\|_\infty.$$

*Proof.* By well-known Hadamard's inequality, $|\det \mathbf{A}| \leq \prod_{j=1}^{n} \|\mathbf{a}_j\|_2$. And obviously $\|\mathbf{a}_j\|_2 \leq n^{1/2}\|\mathbf{a}_j\|_\infty$. □

**Lemma 4.** *If $(x_1, \ldots, x_n)$ is a vertex of $P$, then for each $j = 1, \ldots, n$ we have $x_j = 0$ or $M^{-1} \leq x_j \leq M$ where*

$$M = m^{m/2} \prod_{j=1}^{n+1} B_j \tag{8a}$$

$$B_j = \max\{1, |a_{1j}|, \ldots, |a_{mj}|\}, \quad j = 1, \ldots, n \tag{8b}$$

$$B_{n+1} = \max\{1, |b_1|, \ldots |b_m|\}. \tag{8c}$$

*Proof.* It is well-known from the simplex algorithm that every vertex $\mathbf{x}$ of the polyhedron is a solution of a system $\mathbf{A}'\mathbf{x}' = \mathbf{b}'$, where $\mathbf{x}' = (x'_1, x'_2, \ldots)$ are the non-zero components of $\mathbf{x}$, $\mathbf{A}'$ is a non-singular submatrix of $\mathbf{A}$, and $\mathbf{b}'$ is a subvector of $\mathbf{b}$. By Cramer's rule we have $x'_j = (\det \mathbf{A}'_j)/(\det \mathbf{A}')$ where $\mathbf{A}'_j$ denotes $\mathbf{A}'$ with the $j$-th column replaced by $\mathbf{b}'$. Because $m \leq n$, Lemma 3 implies $|\det \mathbf{A}'_j|, |\det \mathbf{A}'| \leq M$, which proves the claim. □

**Lemma 5.** *If the polyhedron $P$ is bounded then for every $\mathbf{x} \in P$, each side of equation (7) is not greater than*

$$N = M \sum_{j=1}^{n+1} B_j. \tag{9}$$

*Proof.* Every $(x_1, \ldots, x_n) \in P$ is a convex combination of the vertices of $P$, hence, by Lemma 4, each $x_j$ satisfies $x_j \leq M$. We have $|a_{ij}| \leq B_j$ and $|b_i| \leq B_{n+1}$. There are at most $n$ summands on each side of equation (7). □

## 4. Encoding

In this section we prove Theorem 1 by giving a linear-time algorithm to encode the polyhedron $P$ as a face of a local marginal polytope. We assume here that $P$ is bounded—we relax this requirement later in §5.

The input of the algorithm is a set of equations (7). Its output is a min-sum problem $(V, E, K, \mathbf{g})$ with $|K| = 3$ labels and with weights $g_u(k) = 0$ for all $u \in V$ and $k \in K$, and $g_{uv}(k,\ell) \in \{0, 1\}$ for all $\{u, v\} \in E$ and $k, \ell \in K$. This min-sum problem is such that $\min_{\boldsymbol{\mu}\in\Lambda} \langle \mathbf{g}, \boldsymbol{\mu} \rangle = 0$ if and only if $P \neq \emptyset$. Thus, the case $P = \emptyset$ is indicated by $\min_{\boldsymbol{\mu}\in\Lambda} \langle \mathbf{g}, \boldsymbol{\mu} \rangle > 0$. It further means that if $P \neq \emptyset$ then for every $\boldsymbol{\mu} \in \Lambda^*(\mathbf{g})$ we inevitably have $\mu_{uv}(k,\ell) = 0$ whenever $g_{uv}(k,\ell) = 1$.

It will be convenient to depict min-sum problems by pictures, similarly as *e.g.* in [11]. Figure 4 illustrates the meaning of constraints (3) in these pictures. In the sequel, only a subset of the nine edges between two objects are shown, where the visible edges have weight $g_{uv}(k,\ell) = 0$ and the invisible edges have weight $g_{uv}(k,\ell) = 1$.

### 4.1. Elementary constructions

The encoding algorithm uses several elementary constructions as its building blocks. Each construction is a standalone min-sum problem that imposes a certain simple constraint on some unary pseudomarginals of every optimal solution:

COPY, Figure 2(a), enforces equality of two unary pseudomarginals $a, d$ in two objects $\{u, v\} \in E$ while imposing no other constraints on $b, c, e, f$. Precisely, if $a, b, c, d, e, f \geq 0$ and $a + b + c = 1 = d + e + f$, then there exist pairwise pseudomarginals satisfying (3) if and only if $a = d$.

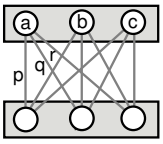


Figure 1. A pair of objects $\{u, v\} \in E$ with $|K| = 3$ labels. Objects $u \in V$ are depicted as boxes, labels $k \in K$ as nodes, and label pairs $(k, \ell) \in K \times K$ as edges. Each node is assigned a unary pseudomarginal $\mu_u(k)$ and each edge is assigned a pairwise pseudomarginal $\mu_{uv}(k, \ell)$. One constraint (3b) imposes for unary pseudomarginals $a, b, c$ that $a + b + c = 1$. One constraint (3a) imposes for pairwise pseudomarginals $p, q, r$ that $a = p + q + r$.

ADDITION, Figure 2(b), adds two unary pseudomarginals $a, b$ in one object and represents the result as a unary pseudomarginal $c = a+b$ in another object. No other constraints are imposed on the remaining unary pseudomarginals.

EQUALITY, Figure 2(c), enforces equality of two unary pseudomarginals $a, b$ in a single object, introducing two auxiliary objects. No other constraints are imposed on the remaining unary pseudomarginals. In the sequel, this construction will be abbreviated by omitting the two auxiliary objects and writing the equality sign between the two nodes, as in Figure 2(d).

POWERS, Figure 2(e), creates the sequence of unary pseudomarginals with values $2^i a$ for $i = 0, \ldots, d$, each in a separate object. We will call $d$ the *depth* of the pyramid.

NEGPOWERS, Figure 2(f), is similar to POWERS but constructs values $2^{-i}$ for $i = 0, \ldots, d$.

Figure 3 shows an example of how the elementary constructions can be combined.

### 4.2. The algorithm

Now we describe the whole encoding algorithm. In the algorithm, labels and objects are numbered by integers, $K = \{1, 2, 3\}$ and $V = \{1, \ldots, |V|\}$.

The algorithm is initialized as follows:

1. For each variable $x_j$ in (5), introduce a new object $j$ into $V$. The variable $x_j$ will be represented by pseudomarginal $\mu_j(1)$. After this step, we have $|V| = n$.
2. For each such object $j \in V$, build POWERS to the depth $d_j = \lfloor \log_2 B_j \rfloor$ based on label 1. This yields the sequence of numbers $2^i \mu_j(1)$ for $i = 0, \ldots, d_j$.
3. Build NEGPOWERS to the depth $d = \lceil \log_2 N \rceil$. The number $2^{-d}$ will play the role of the unit in our construction, it is the scale mentioned in Theorem 1. The choice of $d$ ensures that all values that have to be represented by pseudomarginals will be bounded by 1.

Then the algorithm proceeds by encoding each equation (7) in turn. The $i$-th equation is encoded as follows:

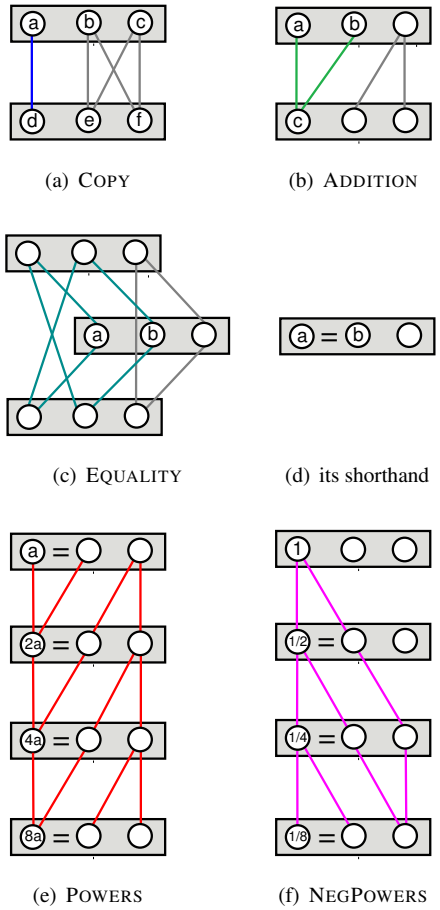


Figure 2. Elementary constructions.

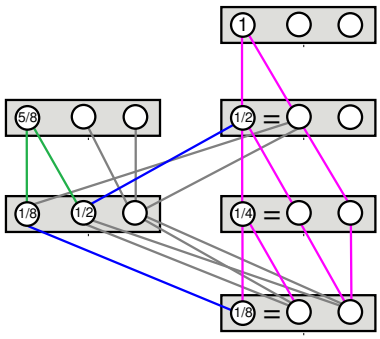


Figure 3. Construction of value $\frac{5}{8}$. The edge colors distinguish different elementary constructions. The example can be generalized in an obvious way to construct the value $2^{-d}k$ for any $d, k \in \mathbb{N}$ such that $2^{-d}k \leq 1$. If more than two values are added, intermediate results are stored in auxiliary objects using COPY.

1. Construct pseudomarginals with values $a_{ij}^{+} x_j$, $a_{ij}^{-} x_j$ by summing selected values from the powers built in Step 2 of the initialization, similarly as in Figure 3.
2. Construct a pseudomarginal with value $2^{-d} b_i$ by summing selected values from the negative powers built in

Step 3 of the initialization, similarly as in Figure 3. The value $2^{-d}b_i$ represents $b_i$, which sets the scale between the input and output polytope to $2^{-d}$.

3. Represent each side of the equation by summing all its terms by repetitively applying ADDITION and COPY.
4. Apply COPY to enforce equality of the two sides of the equation.

When the algorithm finishes, the output min-sum problem encodes the (nonempty) input polytope as

$$P = \pi(\Lambda^*(\mathbf{g})) \tag{10}$$

where $\pi\colon \mathbb{R}^I \to \mathbb{R}^n$ is the scaled coordinate-erasing projection given by

$$(x_1, \dots, x_n) = \pi(\boldsymbol{\mu}) = 2^d(\mu_1(1), \dots, \mu_n(1)). \tag{11}$$

Figure 4 shows the output min-sum problem for an example polytope $P$.

### 4.3. The length of the encoding

Here we finish the proof of Theorem 1 by showing that the encoding time is linear in the length $L$ of the description of the input polyhedron $P$. Since this time is obviously[1] linear in $|E|$, it suffices to show that $|E| = \mathcal{O}(L)$.

Object pairs are created only when an object is created and the number of object pairs added with one object is always bounded by a constant. Therefore $|E| = \mathcal{O}(|V|)$.

Let $P$ be described by a bit sequence including the binary representations of all entries of $\mathbf{A}$ and $\mathbf{b}$. The length of this sequence is $L$. Clearly,

$$L \geq \max\Big\{mn,\ \sum_{j=1}^{n+1} \log_2 B_j\Big\}. \tag{12}$$

The algorithm initialization creates $\sum_{j=1}^{n}(d_j + 1)$ objects via POWERS and $d + 1$ objects via NEGPOWERS. By comparison with (12), both these numbers are $\mathcal{O}(L)$.

Finally, encoding one equality (7) adds at most as many objects as there are bits in the binary representation of all its coefficients. The cumulative sum is thus $\mathcal{O}(L)$.

## 5. Reducing a linear program to linear optimization over a local marginal polytope

In this section we show how to efficiently reduce any linear program to linear optimization over a local marginal polytope. By saying that *problem $A$ can be reduced to problem $B$* we mean there is an oracle to solve problem $B$ which takes constant time and which can be called (possibly repeatedly) to solve problem $A$. We further assume that if problem $B$ is a linear program then the oracle returns both an optimal argument and the optimal value.

The input linear program is assumed to have the form

$$P^*(\mathbf{c}) = \operatorname*{argmin}_{\mathbf{x}\in P} \langle \mathbf{c}, \mathbf{x}\rangle \tag{13}$$

where $P$ is given by (5) and $\mathbf{c} = (c_1, \dots, c_n) \in \mathbb{Z}^n$.

The encoding in §4 can be applied only to a bounded polyhedron $P$ but the LP (13) can be unbounded. This issue is settled by the following lemma, which is not surprising but proved here for completeness.

**Lemma 6.** *Every linear program (13) can be reduced in linear time to a linear program over a bounded polyhedron.*

*Proof.* Denote $H(\alpha) = \{\, \mathbf{x} \in \mathbb{R}^n \mid \langle \mathbf{1}, \mathbf{x}\rangle \leq \alpha \,\}$. By Lemma 4, all vertices of $P$ are contained in the halfspace $H(nM)$. Clearly,

$$\min_{\mathbf{x}\in P\cap H(nM)} \langle \mathbf{c}, \mathbf{x}\rangle \ \geq \ \min_{\mathbf{x}\in P\cap H(2nM)} \langle \mathbf{c}, \mathbf{x}\rangle. \tag{14}$$

Each side of (14) is a linear program over a bounded polyhedron. Inequality (14) is tight if and only if (13) is bounded, in which case (14) has the same optimum as (13). The linear programs (14) are infeasible if and only if (13) is infeasible.

The description length of numbers $nM$ and $2nM$ is $\mathcal{O}(L)$, thus the reduction is done in linear time. □

By Lemma 6 and Theorem 1, any linear program (13) can be reduced in linear time to optimizing a linear function over a face of $\Lambda$. Given an oracle to optimize a linear function over $\Lambda$, it may seem unclear how to optimize a linear function over a *face* of $\Lambda$. But this can be done by setting some pairwise weights to a large constant, $g_\infty$.

Precisely, let $(V, E, K, \mathbf{g}')$ be the min-sum problem encoding $P$, as constructed in §4. Define the vector $\mathbf{g} \in \overline{\mathbb{R}}^I$ by

$$g_i(k) = \begin{cases} c_i & \text{if } k = 1 \text{ and } i \in \{1, \dots, n\}, \\ 0 & \text{otherwise,} \end{cases} \tag{15a}$$

$$g_{ij}(k, \ell) = \begin{cases} 0 & \text{if } g'_{ij}(k, \ell) = 0, \\ g_\infty & \text{if } g'_{ij}(k, \ell) = 1. \end{cases} \tag{15b}$$

The constant $g_\infty$ must be large enough to ensure that every $\boldsymbol{\mu} \in \Lambda^*(\mathbf{g})$ satisfies $\mu_{ij}(k, \ell) = 0$ whenever $g_{ij}(k, \ell) = g_\infty$, so that $\Lambda^*(\mathbf{g}) \subseteq \Lambda^*(\mathbf{g}')$. Then

$$P^*(\mathbf{c}) = \pi(\Lambda^*(\mathbf{g})) \tag{16}$$

because by (15a), the components of $\boldsymbol{\mu}$ multiplied by $c_i$ in the product $\langle \mathbf{g}, \boldsymbol{\mu}\rangle$ are precisely those that represent the variables $x_i$ of the input problem (13). We assume here that $P$

[1] The only thing that may not be obvious is how to multiply large integers $a, b$ in linear time. But this issue can be avoided by instead computing $p(a, b) = 2^{\lceil \log_2 a\rceil + \lceil \log_2 b\rceil}$, which can be done in linear time using bitwise operations. Since $ab \leq p(a, b) \leq (2a)(2b)$, the bounds like $M$ become larger but this does not affect the overall complexity.

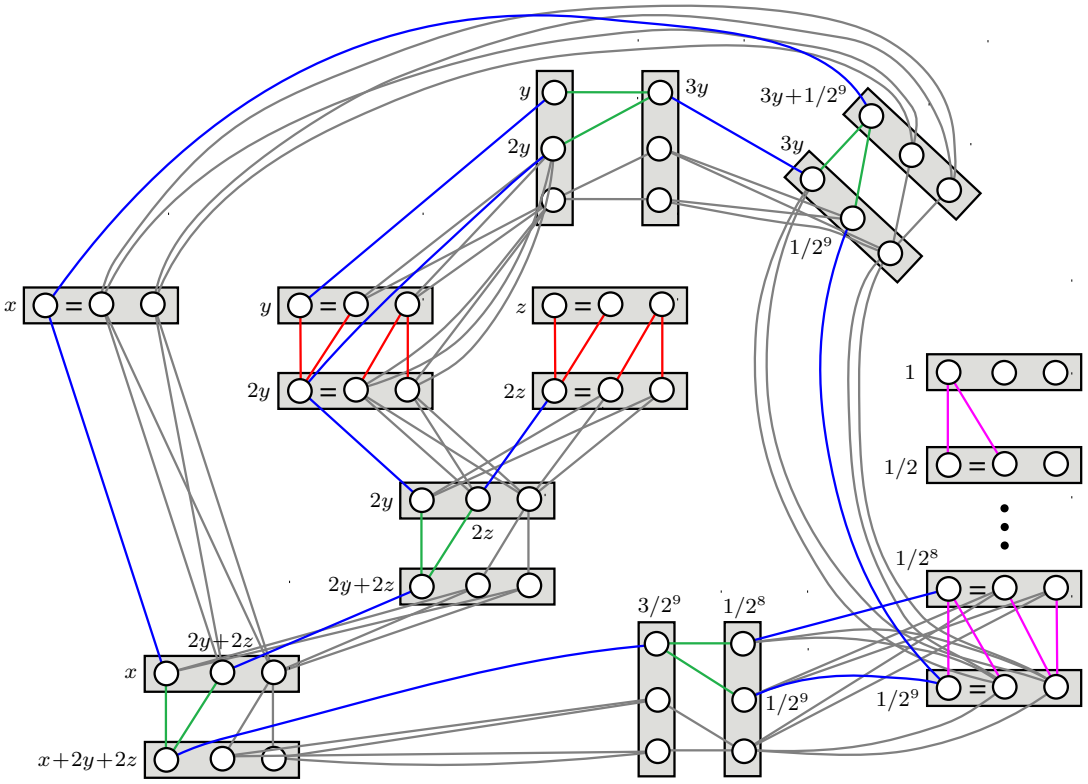


Figure 4. The min-sum problem encoding the polytope $P = \{\,(x, y, z) \mid x + 2y + 2z = 3;\ -x + 3y = -1;\ x, y, z \geq 0\,\}$.

is bounded and non-empty; recall that the case $P = \emptyset$ is indicated by $\min_{\boldsymbol{\mu} \in \Lambda} \langle \mathbf{g}', \boldsymbol{\mu} \rangle > 0$.

It remains to choose $g_\infty$. The situation is different depending on whether we are allowed to use infinite weights (components of $\mathbf{g}$) or not. If infinite weights are allowed, we simply set $g_\infty = \infty$. This proves Theorem 2.

If infinite weights are not allowed, $g_\infty$ must be a large enough finite constant. Unfortunately, manipulation with these large numbers increases the complexity of the reduction. This is given by Theorem 8. To prove it, we need Lemma 7, which refines Lemma 4 for the special case of the local marginal polytope.

**Lemma 7.** *Let $\Lambda$ be a local marginal polytope given by a triplet $(V, E, K)$ where $|K| = 3$. Every component $\mu$ of every vertex $\boldsymbol{\mu}$ of $\Lambda$ satisfies $\mu = 0$ or $2^{-|V|-6|E|} \leq \mu$.*

*Proof.* Write constraints (3a) and (3b) as $\mathbf{U}\boldsymbol{\mu} = \mathbf{u}$. The matrix $\mathbf{U}$ has $|V| + 2|K||E| = |V| + 6|E|$ rows and $|K||V| + |K|^2|E| = 3|V| + 9|E|$ columns. The composed matrix $[\mathbf{U} \mid \mathbf{u}]$ contains exactly 4 non-zero elements in every row, each of them being either $-1$ or 1. Let $\mathbf{U}'$ be a non-singular submatrix of $[\mathbf{U} \mid \mathbf{u}]$. By Hadamard's inequality,

$$|\det \mathbf{U}'| = |\det \mathbf{U}'^{\top}| \leq \prod_{j=1}^{|V|+6|E|} \sqrt{4} = 2^{|V|+6|E|},$$

which implies the claimed lower bound. □

**Theorem 8.** *Any linear program (13) can be reduced in time and space $\mathcal{O}(L(L + L'))$ to a linear optimization (allowing only finite weights) over a local marginal polytope with 3 labels, where $L'$ is the length of the binary representation of $\mathbf{c}$.*

*Proof.* Let $(V, E, K, \mathbf{g}')$ be the min-sum problem encoding $P$, where we assume that $P$ is bounded and non-empty. Let $\mathbf{g}$ be given by (15) where

$$g_\infty = 1 + C2^{|V|+6|E|+1}, \qquad C = \sum_{i=1}^{n} |c_i|.$$

We claim that then (16) holds. To prove this, we need to show that every $\boldsymbol{\mu} \in \Lambda^*(\mathbf{g})$ satisfies $\mu_{ij}(k, \ell) = 0$ whenever $g_{ij}(k, \ell) = g_\infty$. It suffices to show this only for the vertices of $\Lambda^*(\mathbf{g})$ because taking a convex combination cannot violate the condition $\mu_{ij}(k, \ell) = 0$.

Since $P$ is non-empty, we have $\min_{\boldsymbol{\mu} \in \Lambda} \langle \mathbf{g}', \boldsymbol{\mu} \rangle = 0$. The constant $C$ is chosen such that $\min_{\boldsymbol{\mu} \in \Lambda} \langle \mathbf{g}, \boldsymbol{\mu} \rangle \leq C$. For contradiction, suppose for some $\boldsymbol{\mu} \in \Lambda^*(\mathbf{g})$ there is some $\{(i, k), (j, \ell)\}$ such that $g_{ij}(k, \ell) = g_\infty$ and $\mu_{ij}(k, \ell) > 0$. But by Lemma 7, $\mu_{ij}(k, \ell) > 2^{-|V|-6|E|}$. Thus,

$$\min_{\boldsymbol{\mu} \in \Lambda} \langle \mathbf{g}, \boldsymbol{\mu} \rangle \geq g_\infty 2^{-|V|-6|E|} - C > C.$$

The binary length of $g_\infty$ is $\mathcal{O}(L + L')$. It occurs in $\mathbf{g}$ at $\mathcal{O}(|K|^2|E|) = \mathcal{O}(L)$ positions, thus the binary length of $\mathbf{g}$ is $\mathcal{O}(L(L + L'))$. This induces the claimed time and space complexities. □

Recall that an algorithm runs in a strongly polynomial time if its number of operations in the arithmetic model of computation (in which any operation takes the unit time) is bounded by a polynomial in the number of integers in the input instance and the space used by the algorithm is bounded by a polynomial in the size of the input. Our last theorem specifies a class of linear programs that can be reduced in strongly polynomial time to the LP relaxation of a min-sum problem.

**Theorem 9.** *Every linear program (13) where* $\mathbf{A} \in \{-1, 0, 1\}^{m\times n}$ *and* $\mathbf{b} \in \{-1, 0, 1\}^m$ *can be reduced in strongly polynomial time to a linear optimization over a local marginal polytope.*

*Proof.* In this case $L = \mathcal{O}(mn)$. Bounding the linear program (13) by Lemma 6 adds (twice) an equation whose binary length is $\mathcal{O}(n + m \log m)$. In the arithmetic model of computation, $g_\infty$ from Theorem 8 can be computed using $\mathcal{O}(mn)$ additions and multiplications. Furthermore, all the computed numbers are represented in space $\mathcal{O}(mnL)$. □

## 6. Consequences

Our result has a number of immediate consequences.

Most importantly, it shows that solving the LP relaxation of a pairwise min-sum problem is comparably hard as solving any LP. This is straightforward if infinite weights are allowed. Then, by Theorem 2, the reduction is done in time $\mathcal{O}(L)$, while the best known algorithm [5] for general LP has time complexity $\mathcal{O}(n^{3.5}L^2 \log L \log \log L)$. Finding a very fast algorithm, such as $\mathcal{O}(L^2 \log L)$, to solve LP relaxation of min-sum problems (which permit infinite weights) would imply improving the best-known complexity of LP.

Our result makes more precise the known observation that local marginal polytopes with $|K| = 3$ labels are more complex than those with 2 labels. Any pairwise min-sum problem with 2 labels can be reduced in linear time to a quadratic pseudoboolean optimization problem, whose LP relaxation can be reduced in linear time to a max-flow problem [1, 8], which has a lower best known complexity than a general LP. Local marginal polytopes with 2 labels have half-integral vertices (*i.e.*, all components of the vertices are in $\{0, \frac{1}{2}, 1\}$) [6, 11], while the components of the vertices of local marginal polytopes with 3 labels can have much more general values, as indicated by Figure 4. Moreover, there seems to be not much difference in complexity between local marginal polytopes with 3 labels and those with 4 or more labels.

When solving the LP relaxation of a min-sum problem by the simplex algorithm, due to high degeneracy of the local marginal polytope the simplex algorithm sometimes stays in a single basic solution for a very large number of iterations, only changing degenerate bases (this is known as *stalling*). Finding a pivoting rule that would guarantee no stalling would imply this rule is applicable to any LP.

Last but not least, the question whether there is a strongly polynomial algorithm for the LP relaxation of the min-sum problem is shown to be equivalent to the question whether any LP with components of $\mathbf{A}$ and $\mathbf{b}$ in $\{-1, 0, 1\}$ (without any restriction on $\mathbf{c}$) has a strongly polynomial algorithm.

## Acknowledgment

Both authors were supported by the Grant Agency of the Czech Republic project P202/12/2071. Besides, DP was supported by the EC project FP7-ICT-247525 HUMAVIPS and TW by the EC project FP7-ICT-270138 DARWIN.

## References

[1] E. Boros and P. L. Hammer. Pseudo-Boolean optimization. *Discrete Applied Mathematics*, 123(1-3):155–225, 2002. 6

[2] 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*, pages 109–118, 2001. 1

[3] D. Cohen, M. Cooper, P. Jeavons, and A. Krokhin. The complexity of soft constraint satisfaction. *Artificial Intelligence*, 170:983–1016, 2006. 1

[4] J. A. De Loera and S. Onn. All linear and integer programs are slim 3-way transportation programs. *SIAM J. on Optimization*, 17(3):806–821, 2006. 1

[5] N. Karmarkar. A new polynomial-time algorithm for linear programming. In *Proceedings of the sixteenth annual ACM symposium on Theory of computing*, STOC '84, pages 302–311, New York, NY, USA, 1984. ACM. 6

[6] V. Kolmogorov. Convergent tree-reweighted message passing for energy minimization. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 28(10):1568–1583, 2006. 6

[7] A. Koster, C. van Hoesel, and A. Kolen. The partial constraint satisfaction problem: Facets and lifting theorems. *Operations Research Letters*, 23(3–5):89–97, 1998. 1

[8] C. Rother, V. Kolmogorov, V. S. Lempitsky, and M. Szummer. Optimizing binary MRFs via extended roof duality. In *Conf. Computer Vision and Pattern Recognition (CVPR)*, 2007. 6

[9] M. I. Shlezinger. Syntactic analysis of two-dimensional visual signals in noisy conditions. *Cybernetics and Systems Analysis*, 12(4):612–628, 1976. Translation from Russian. 1

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

[11] T. Werner. A linear programming approach to max-sum problem: A review. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 29(7):1165–1179, 2007. 2, 6