LP Relaxation of the Potts Labeling Problem Is as Hard as Any Linear Program

Daniel Průša and Tomáš Werner

Abstract—In our recent work, we showed that solving the LP relaxation of the pairwise min-sum labeling problem (also known as MAP inference in graphical models or discrete energy minimization) is not much easier than solving any linear program. Precisely, the general linear program reduces in linear time (assuming the Turing model of computation) to the LP relaxation of the min-sum labeling problem. The reduction is possible, though in quadratic time, even to the min-sum labeling problem with planar structure. Here we prove similar results for the pairwise min-sum labeling problem with attractive Potts interactions (also known as the uniform metric labeling problem).

Index Terms—Markov random field, graphical model, MAP inference, discrete energy minimization, valued constraint satisfaction, linear programming relaxation, uniform metric labeling problem, Potts model

1 Introduction

The *pairwise min-sum (labeling) problem* consists in minimizing a sum of unary and binary (also called pairwise) cost functions of discrete variables. It is also known as (pairwise) discrete energy minimization [1], [2], valued constraint satisfaction [3], or MAP inference in graphical models [4]. It has many applications in computer vision, machine learning, and other fields. This NP-hard problem has a natural linear programming (LP) relaxation [4], [5], [6], [7], [8], which underlies many successful algorithms to tackle the problem (see [2] and the references therein). Therefore it would have a great practical impact to have efficient algorithms to solve this LP relaxation. The popular simplex and interior point methods are prohibitively inefficient for large instances of the LP relaxation, often arising, e.g., in computer vision. Our recent work [9] showed that, unfortunately, solving the LP relaxation of the pairwise min-sum problem with three variable states (labels) is as hard as solving the general linear program. Precisely, the latter reduces to the former in linear time, assuming the Turing model of computation. Therefore it is unlikely that a very efficient algorithm for the LP relaxation exists.

This negative result suggests the question whether there are any interesting subclasses of the min-sum problem for which the LP relaxation is easier than the general LP and thus there is a hope for efficient algorithms. One such subclass is the pairwise min-sum problem with two labels, for which the LP relaxation has half-integral solutions and reduces in linear time to max-flow [10], [11]. Thus the LP relaxation can be solved very efficiently because the complexity of best known algorithms for max-flow is much better than for the general LP.

Another subclass is the *metric labeling problem* [12], [13], [14], a pairwise min-sum problem in which the pairwise cost functions satisfy the axioms of a metric. An important special case is the *uniform metric*, in statistical physics known as the *attractive Potts interaction*. We refer to the pairwise min-sum problem with attractive Potts interactions as the *Potts (labeling) problem*. The LP relaxation for this (still NP-hard) problem was proposed in [13] and later generalized to any metric in [14]. For the Potts problem, the LP relaxation [4], [5], [6], [7], [8] coincides with that in [13], [14].

The LP relaxation is the basis for approximation algorithms to the metric labeling problem with theoretical approximation guarantees, in particular for the uniform metric where the approximation ratio is most favorable [13], [14], [15]. There is another class of approximation algorithms for metric labeling problems, α-expansion algorithms [12], which call a max-flow solver a small number of times and thus they are very efficient. They achieve comparable worst-case approximation guarantees [16] but the algorithms based on LP relaxation are often more accurate in practice [1], [2]. Moreover, for the multiway cut problem, closely related to the Potts labeling problem, the LP relaxation is the only known way to achieve the best possible approximation [17].

In this article, we show that solving the LP relaxation is hard even for the Potts labeling problem. Precisely, the general linear program can be reduced in linear time to the LP relaxation of the Potts labeling problem with three labels (Theorem 4). Unlike in [9] where the input LP is directly encoded by a min-sum problem, we proceed in a different way. By duality, the LP problem is linear-time equivalent to the linear feasibility (LF) problem (Lemma 3), therefore it suffices to construct a reduction from LF. We do this in two steps: first LF with rational coefficients is reduced to LF with coefficients in $\{-1,0,1\}$ by algebraic manipulations (Section 3) and then this problem is reduced to the LP relaxation of the Potts problem (Section 4).

This construction allows us to strengthen the result from [9] for general min-sum problem because infinite costs are no longer needed to achieve linear time. It allows us to formulate several other results. As in [9], the reduction has a polyhedral formulation (Theorem 5): any polytope is linear-time representable as a face of the feasible set of the LP relaxation [13] of a Potts problem, which we call the *relaxed Potts polytope*. We show (Theorem 8) that the reduction to the LP relaxation of the Potts problem can be also understood as a reduction to the LP relaxation of the *multiway cut problem* [17]. Finally, again similarly to [9], the reduction can be modified such that the output Potts problem is planar, but this needs more than linear time (Theorem 9).

2 LP Relaxation of Min-Sum Problem

The pairwise min-sum (labeling) problem is defined as

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

where (V,E) is a graph with V a finite set of *objects* and $E\subseteq\binom{V}{2}$ a set of *object pairs*, K is a finite set of *labels*, and $g_u\colon K\to\mathbb{R}$ and $g_{uv}\colon K\times K\to\mathbb{R}$ are unary and pairwise *cost functions*, adopting that $g_{uv}(k,\ell)=g_{vu}(\ell,k)$.

The LP relaxation of this problem reads

$$\min_{\mu\in\Lambda}\langle \mathbf{g},\boldsymbol{\mu}\rangle, \tag{2}$$

where $\mathbf{g}\in\mathbb{R}^I$ and $\boldsymbol{\mu}\in\mathbb{R}^I$ is the vector with components $g_u(k), g_{uv}(k,\ell)$ and $\mu_u(k),\mu_{uv}(k,\ell)$, respectively, and

$$I=(V\times K)\cup\{\,\{(u,k),(v,\ell)\}\mid \{u,v\}\in E,\ k,\ell\in K\,\}.$$

The set $\Lambda\subseteq\mathbb{R}^I$ contains all vectors $\boldsymbol{\mu}\geq\mathbf{0}$ satisfying

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

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

• *The authors are with the Department of Cybernetics, Faculty of Electrical Engineering, Czech Technical University, Karlovo náměstí 13, Praha 12135, Czech Republic. E-mail: {prusapa1, werner}@fel.cvut.cz.*

Manuscript received 10 Feb. 2016; revised 3 June 2016; accepted 6 June 2016. Date of publication 19 June 2016; date of current version 12 June 2017.
Recommended for acceptance by S. Nowozin.
For information on obtaining reprints of this article, please send e-mail to: reprints@ieee.org, and reference the Digital Object Identifier below.
Digital Object Identifier no. 10.1109/TPAMI.2016.2582165

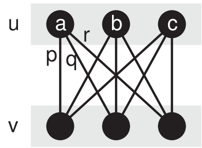

Fig. 1. One object pair $\{u,v\} \in E$ with $|K|=3$ labels. Objects $u,v \in V$ are depicted as boxes, labels $(u,k) \in I$ as nodes, and label pairs $\{(u,k),(v,\ell)\} \in I$ as edges. Note the meaning of constraints (3): for unary pseudomarginals a,b,c and pairwise pseudomarginals p,q,r, equality (3a) reads $a=p+q+r$ and equality (3b) reads $a+b+c=1$.

where $N_u = \{\, v \mid \{u,v\} \in E \,\}$ denotes the neighbors of object u. Following [4], we refer to Λ as the *local marginal polytope* and to $\mu_u(k), \mu_{uv}(k,\ell)$ as *pseudomarginals*. The meaning of constraints 3a is illustrated in Fig. 1.

A *reparameterization* of a cost vector $\mathbf{g} \in \mathbb{R}^I$ is a cost vector $\mathbf{g}' \in \mathbb{R}^I$ given by

$$g'_u(k) = g_u(k) - \sum_{v \in N_u} \varphi_{uv}(k) \tag{4a}$$

$$g'_{uv}(k,\ell) = g_{uv}(k,\ell) + \varphi_{uv}(k) + \varphi_{vu}(\ell), \tag{4b}$$

where $\varphi_{uv}(k) \in \mathbb{R}$ ($u \in V$, $v \in N_u$, $k \in K$). Reparameterizations preserve $\langle \mathbf{g}, \boldsymbol{\mu} \rangle$ for every $\boldsymbol{\mu}$ satisfying (3).

2.1 Potts Labeling Problem

Problem (1) in which pairwise cost functions g_{uv} satisfy metric axioms has been called the *metric labeling problem* [12], [13], [14], [15]. Its special case is obtained for the *uniform metric* (the *attractive Potts interaction*)

$$g_{uv}(k,\ell) = h_{uv} [\![k \neq \ell]\!], \tag{5}$$

where $h_{uv} \geq 0$, and $[\![k \neq \ell]\!] = 1$ if $k \neq \ell$ and $[\![k \neq \ell]\!] = 0$ if $k = \ell$. We refer to problem (1) with pairwise costs (5) as the *Potts (labeling) problem*.

In this case, problem (2) can be simplified [14] by minimizing out the pairwise pseudomarginals μ_{uv}. For a fixed object pair $\{u,v\} \in E$, minimizing $\langle g_{uv}, \mu_{uv} \rangle$ over $\mu_{uv} \geq 0$ subject to (3a) is a discrete transportation problem with transport costs g_{uv}. If g_{uv} has the form (5), the optimal value of this problem is given explicitly as $\frac{1}{2} h_{uv} \sum_{k \in K} |\mu_u(k) - \mu_v(k)|$. Therefore (2) is equivalent to minimizing

$$\sum_{u \in V} \sum_{k \in K} g_u(k) \mu_u(k) + \sum_{\{u,v\} \in E} \frac{1}{2} h_{uv} \sum_{k \in K} |\mu_u(k) - \mu_v(k)|, \tag{6}$$

over unary pseudomarginals $\mu_u(k) \geq 0$ subject to (3b). This is the relaxation of the Potts problem proposed by Kleinberg and Tardos [13]. It can be written also as

$$\min_{\nu \in \Pi} \langle \mathbf{h}, \nu \rangle, \tag{7}$$

where $\mathbf{h} \in \mathbb{R}^{(V \times K) \cup E}$ is the vector with components $h_u(k) = g_u(k)$ and h_{uv}, and Π is the set of all vectors $\nu \in [0,1]^{(V \times K) \cup E}$ with components $\nu_u(k), \nu_{uv}$ satisfying

$$\sum_{k \in K} |\nu_u(k) - \nu_v(k)| \leq 2\nu_{uv}, \quad \{u,v\} \in E, \tag{8a}$$

$$\sum_{k \in K} \nu_u(k) = 1, \quad u \in V. \tag{8b}$$

We will refer to Π as the *relaxed Potts polytope*.

3 Input Polyhedron

Our key construction in the paper will be a linear-time representation of any convex polyhedron as the optimal set of the LP relaxation of a Potts problem. We assume the input polyhedron in the form

$$P = \{\, \mathbf{x} \in \mathbb{R}^n \mid \mathbf{A}\mathbf{x} = \mathbf{0},\ x_n = 1,\ \mathbf{x} \geq \mathbf{0} \,\}, \tag{9}$$

where $\mathbf{A} = [a_{ij}] \in \mathbb{Q}^{m \times n}$ and x_n denotes the last component of the vector $\mathbf{x} = (x_1, \dots, x_n)$. Note that the equation $x_n = 1$ makes the homogeneous linear system $\mathbf{A}\mathbf{x} = \mathbf{0}$ non-homogeneous, with the right-hand side being the negative last column of $\mathbf{A}$. Each row and column of $\mathbf{A}$ is assumed to have at least one non-zero.

By 'linear time' we mean time $O(N)$ where N is the size of the input, i.e., the number of bits needed to encode matrix $\mathbf{A}$ in binary. That is, we assume the Turing model of computation. Let us define the size of a matrix precisely. For a scalar $a \in \mathbb{Q}$, we define

$$\operatorname{size}(a) = \log_2(|pq| + 1), \tag{10}$$

where $p, q \in \mathbb{Z}$ are such that $a = p/q$ assuming that q does not divide p unless $q = 1$ or $p = 0$. For a matrix $\mathbf{A} \in \mathbb{Q}^{m \times n}$, we define

$$\operatorname{size}(\mathbf{A}) = \sum_{j=1}^{n} \sum_{i=1}^{m} \operatorname{size}(a_{ij}). \tag{11}$$

As $\operatorname{size}(a) = 0$ for $a = 0$, (11) underestimates the true size of matrix $\mathbf{A}$ by neglecting the space needed, e.g., for storing the indices of zero entries. This does not matter because if the time of an algorithm is linear in $\operatorname{size}(\mathbf{A})$, it is at most linear in the true size of $\mathbf{A}$. On the contrary, not counting zero entries makes our results stronger because it allows for a sparse representation of $\mathbf{A}$.

In the rest of this section, we transform the description (9) of the input polyhedron by algebraic manipulations to a form suitable for encoding by a Potts problem.

3.1 From Rationals to Integers

First, the homogeneous linear system $\mathbf{A}\mathbf{x} = \mathbf{0}$ in (9) with rational coefficients is transformed to a linear system with integer coefficients.[1] For each non-zero input coefficient $a_{ij} = p_{ij}/q_{ij} \in \mathbb{Q}$ with $p_{ij}, q_{ij} \in \mathbb{Z}$, we create an auxiliary variable y_{ij} and the equation

$$|q_{ij}| y_{ij} = |p_{ij}| x_j. \tag{12}$$

Then in the input system we replace every non-zero term $a_{ij} x_k$ with $\operatorname{sgn}(a_{ij}) y_{ij}$. The size of the output is clearly linear in the size of the input.[2]

Example 1. The system

$$\begin{aligned} \frac{2}{7}x_1 + \frac{3}{5}x_2 - 2x_3 &= 0 \\ \frac{7}{3}x_1 - \frac{1}{2}x_2 \qquad\quad &= 0, \end{aligned}$$

is transformed to the system

$$\begin{array}{ccc} 2x_1 = 7y_{11} & 3x_2 = 5y_{12} & 2x_3 = y_{13} \\ 7x_1 = 3y_{21} & x_2 = 2y_{22} & \\ & y_{11} + y_{12} - y_{13} = 0 & \\ & y_{21} - y_{22} = 0. & \end{array}$$

3.2 From Integers to $\{-1, 0, 1\}$

The system $\mathbf{A}\mathbf{x} = \mathbf{0}$ with integer coefficients $\mathbf{A} \in \mathbb{Z}^{m \times n}$ is now transformed in linear time to a homogeneous system with coefficients in $\{-1, 0, 1\}$.

Instead of the usual $(x_1, \dots, x_n)$, let us name the input variables $(x_{10}, \dots, x_{n0})$. The key idea is similar to [18, Section 3.1]. Suppose

1. In [9] we assumed that the input LP has integer coefficients, in other words, this step was omitted.

2. Note that the most obvious reduction, multiplying each equation by the least common multiple of the denominators, would take more than linear time.

we want to construct a product $a_{ij}x_{j0}$ for some $a_{ij} \in \mathbb{N}$. Create the equation system

$$\begin{aligned} x_{j1} &= x_{j0} + y_{j0} & y_{j0} &= x_{j0} \\ x_{j2} &= x_{j1} + y_{j1} & y_{j1} &= x_{j1} \\ &\vdots & &\vdots \\ x_{j,d_j} &= x_{j,d_j-1} + y_{j,d_j-1} & y_{j,d_j-1} &= x_{j,d_j-1} \end{aligned} \tag{13}$$

The first line of this system enforces $x_{j1} = 2x_{j0}$, the second line enforces $x_{j2} = 2x_{j1}$, etc. Consequently,

$$x_{jk} = 2^k x_{j0}. \tag{14}$$

The product $a_{ij}x_{j0}$ can be now obtained by summing appropriate bits of the binary code of a_{ij}. E.g., $11x_{j0} = x_{j0} + x_{j1} + x_{j3}$ because $11 = 2^0 + 2^1 + 2^3$.

The whole reduction proceeds as follows:

1) For each $j = 1, \ldots, n$, create equation system 13 with $d_j = \lfloor \log_2 \max_{i=1}^m |a_{ij}| \rfloor$.
2) For each $i = 1, \ldots, m$, construct non-zero terms $a_{ij}x_{j0}$, sum them, and equate the result to zero.

It is easy to verify that the number of non-zero output terms is linear in $\mathrm{size}(\mathbf{A})$.

Example 2. The system

$$\begin{aligned} 2x_{10} + 11x_{20} - 3x_{30} + x_{40} &= 0 \\ 3x_{10} + 6x_{20} - 5x_{40} &= 0 \end{aligned}$$

is transformed to the system

$$\begin{gathered} x_{11} = x_{10} + y_{10} \quad y_{10} = x_{10} \\ x_{21} = x_{20} + y_{20} \quad y_{20} = x_{20} \\ x_{22} = x_{21} + y_{21} \quad y_{21} = x_{21} \\ x_{23} = x_{22} + y_{22} \quad y_{22} = x_{22} \\ x_{31} = x_{30} + y_{30} \quad y_{30} = x_{30} \\ x_{41} = x_{40} + y_{40} \quad y_{40} = x_{40} \\ x_{42} = x_{41} + y_{41} \quad y_{41} = x_{41} \\ x_{11} + (x_{20} + x_{21} + x_{23}) - (x_{30} + x_{31}) + x_{40} = 0 \\ (x_{10} + x_{11}) + (x_{21} + x_{22}) - (x_{40} + x_{42}) = 0. \end{gathered}$$

3.3 Scaling

A polyhedron (9) with $\mathbf{A} \in \{-1, 0, 1\}^{m \times n}$ is now scaled down such that all its vertices are contained in the box $[0, \frac{1}{n}]^n$. This ensures that all quantities represented by pseudomarginals fit into the interval $[0, 1]$ (see Sections 4.2 and 5.1).

Lemma 1. *Each vertex* $\mathbf{x}$ *of convex polyhedron 9 with* $\mathbf{A} \in \{-1, 0, 1\}^{m \times n}$ *satisfies* $\mathbf{x} \in [0, M]^n$ *where*

$$M = \prod_{j=1}^{n} \sum_{i=1}^{m} |a_{ij}|. \tag{15}$$

Moreover, $\mathrm{size}(M) = O(\mathrm{size}(\mathbf{A}))$.

Proof. See Lemma 4 and Section 4.3 in [9]. □

By Lemma 1, the polyhedron must be scaled down by the factor nM. This can be conveniently done during the transformation in Section 3.2. Let $\mathbf{A} \in \{-1, 0, 1\}^{m \times n}$ be the output matrix and j the index of the last variable of the input system in Section 3.2. Without scaling, we would set $x_j = 1$. To achieve scaling, set $d_j = \lceil \log_2(nM) \rceil$ and $x_{j,d_j} = 1$. By (14), this yields $x_j = 2^{-d_j} \leq (nM)^{-1}$.

Though the number nM can be big, by Lemma 1 its size, and hence the number of added equations, is $O(\mathrm{size}(\mathbf{A}))$.

To summarize Section 3, polyhedron (9) with rational coefficients has been transformed in linear time to a polyhedron of the same form with coefficients $\{-1, 0, 1\}$ and vertices in $[0, \frac{1}{n}]^n$. More precisely, the input polyhedron is a scaled coordinate-erasing projection of the output polyhedron, where the erased coordinates correspond to the auxiliary variables introduced in Sections 3.1 and 3.2. Here, we call a projection *coordinate-erasing* if it acts by erasing a subset of coordinates.

4 Encoding by Potts Problem

Here we will represent the polyhedron obtained in Section 3 by the LP relaxation of a Potts problem. In fact, the output problem will be a *reparameterized Potts problem*, i.e., a min-sum problem with arbitrary unary costs $g_u(k)$ and pairwise costs (4b) with $g_{uv}(k, \ell)$ given by (5). By moving $\varphi_{uv}(k)$ to the unary costs, such a problem can be reparameterized in linear time to a Potts problem with unary costs (4a) and pairwise costs (5).

4.1 Gadgets

We will construct the output problem by gluing small subproblems, called *gadgets*,[3] which encode simple operations on unary pseudomarginals. Each gadget is a reparameterized Potts problem with unary costs $g_u(k) \in \{0, 1\}$ and pairwise costs

$$g_{uv}(k, \ell) = 2[\![k \neq \ell]\!] + \varphi_{uv}(k) + \varphi_{vu}(\ell), \tag{16}$$

(i.e., we set[4] $h_{uv} = 2$ for all $\{u, v\} \in E$ in (5)) where $\varphi_{uv}(k) \in \{-1, 0, 1\}$. In addition, the costs satisfy

$$\min_{k \in K} g_u(k) = 0, \quad u \in V, \tag{17a}$$

$$\min_{k, \ell \in K} g_{uv}(k, \ell) = 0, \quad \{u, v\} \in E. \tag{17b}$$

Each gadget is designed such that its LP relaxation has zero optimal value. It follows that any $\boldsymbol{\mu} \in \Lambda$ is optimal to (2) if and only if

$$g_u(k)\mu_u(k) = 0, \quad u \in V,\ k \in K, \tag{18a}$$

$$g_{uv}(k, \ell)\mu_{uv}(k, \ell) = 0, \quad \{u, v\} \in E,\ k, \ell \in K, \tag{18b}$$

i.e., whenever a cost is positive then the corresponding pseudomarginal must vanish.

We will define gadgets by diagrams such as in Fig. 1, adopting the following conventions. Each non-zero number $\varphi_{uv}(k)$ is written near node (u, k) on the side of object v, where '+' stands for $\varphi_{uv}(k) = 1$ and '−' for $\varphi_{uv}(k) = -1$. A node (u, k) is black if $g_u(k) = 0$ and white if $g_u(k) = 1$. An edge $\{(u, k), (v, \ell)\}$ is drawn only if $g_{uv}(k, \ell) = 0$ and both of its end-nodes are black, otherwise it is invisible. Fig. 2 shows an example.

We will use the following gadgets, defined in Fig. 3:

- Swap swaps two unary pseudomarginals, one of them zero. Precisely, the LP relaxation of this gadget has zero optimal value if and only if the unary pseudomarginals linked by visible edges are equal and the unary pseudomarginals in the white nodes are zero.
- Permute applies Swap several times to arbitrarily permute all the three unary pseudomarginals, one of them zero. The figure shows one possible permutation.
- Copy copies all the three unary pseudomarginals, one of them zero, from one object to another object.

3. When constructing reductions in complexity theory, a *gadget* is a small instance of the output problem that implements a certain simple functionality of the input problem. In [9] we used the term 'elementary construction' instead of 'gadget'.

4. We could have just as well set $h_{uv} = 1$; we chose $h_{uv} = 2$ only for convenience because then all $\varphi_{uv}(k)$ can be integer.

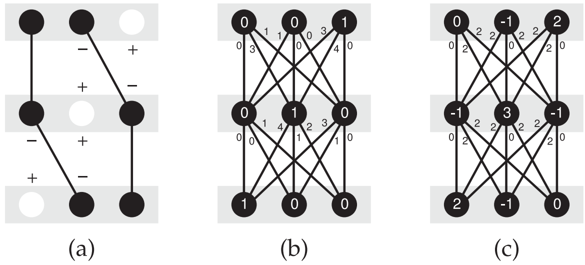

Fig. 2. Our notation for gadgets. (a) shows a gadget in our notation. (b) is the corresponding reparameterized Potts problem with unary costs written inside nodes and pairwise costs written next to edges; each $+$ [$-$] contributes by 1 [-1] to the pairwise cost of the adjacent edge. (c) is the corresponding Potts problem; each $+$ [$-$] contributes by 1 [-1] to the unary cost of the adjacent node, each white node contributes to its unary cost by additional increment 1.

- Unit enforces a unary pseudomarginal to be 1.
- Add1 adds two unary pseudomarginals in a single object and copies the result in another object. The third unary pseudomarginal is copied.
- Add adds two unary pseudomarginals in two different objects. This is done by gluing three Add1gadgets.

Each gadget has several versions obtained by permuting the three labels. Each interface object of Copy, Swap and Add has two black nodes and one white node. This ensures that any versions of Copy and Add can be glued together, possibly after permuting the nodes by Permute. Unit can be glued with any gadget with black node linked to the black node labeled 1. When several gadets are glued, the unary costs in identified nodes (u, k) of their interface objects are summed.

4.2 Encoding

We now describe the encoding algorithm. The input of the algorithm is a polyhedron (9) with $\mathbf{A} \in \{-1, 0, 1\}^{m\times n}$ and the vertices in $[0, \frac{1}{n}]^n$. Its output is a reparameterized Potts problem with $|K| = 3$ labels.

First, we rewrite the system $\mathbf{Ax} = \mathbf{0}$ in (9) as

$$\mathbf{A}^{+}\mathbf{x} = \mathbf{A}^{-}\mathbf{x}, \tag{19}$$

where $a^{+}_{ij} = \max\{a_{ij}, 0\}$ and $a^{-}_{ij} = \max\{-a_{ij}, 0\}$ so that $a^{+}_{ij}, a^{-}_{ij} \in \{0, 1\}$. That is, we have moved negative terms in each equation to the other side of the equation.

Let the three labels of the output problem be named $K = \{1, 2, 3\}$. The encoding proceeds as follows:

1) Set $V = \{1, \dots, n\}$ and $E = \emptyset$. Each variable x_j is now represented by unary pseudomarginal $\mu_j(1)$.
2) For each $i = 1, \dots, m$, encode the ith equation of system (19) as follows:
   a) Construct a unary pseudomarginal equal to the LHS of the equation using Add, permuting labels by Permute if necessary.
   b) Do the same for the RHS.
   c) Equate the LHS and RHS using Copy, permuting labels by Permute if necessary.
3) Encode the equation $x_n = 1$ using Unit.

Assume that the input polyhedron P is bounded (i.e., a polytope). Due to the scaling done in Section 3.3, $\mathbf{A}^{+}\mathbf{x} = \mathbf{A}^{-}\mathbf{x} \le \mathbf{1}$ for all $\mathbf{x} \in P$. Therefore every expression formed in Steps 2a and 2b fits into the feasible interval $[0, 1]$ of pseudomarginals. Recall that the LP relaxation of each gadget has zero optimal value. Since the gadgets are glued such that they encode the input system $\mathbf{Ax} = \mathbf{0}$, the LP relaxation of the output problem will have zero optimal value if and only if P is non-empty. In other words, the output min-sum problem encodes the input polytope as follows:

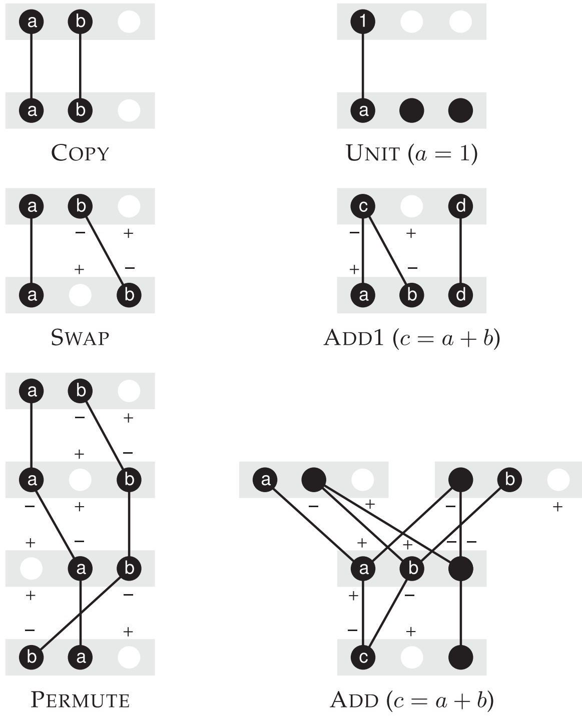

Fig. 3. Potts problems used as gadgets.

- If $P = \emptyset$ then $\min_{\boldsymbol{\mu}\in\Lambda}\langle \mathbf{g}, \boldsymbol{\mu}\rangle > 0$.
- If $P \neq \emptyset$ then $\min_{\boldsymbol{\mu}\in\Lambda}\langle \mathbf{g}, \boldsymbol{\mu}\rangle = 0$ and

$$P = \pi\Big(\operatorname*{arg\,min}_{\boldsymbol{\mu}\in\Lambda}\langle \mathbf{g}, \boldsymbol{\mu}\rangle\Big), \tag{20}$$

where 'argmin' denotes the set of all minimizers and

$$\pi\colon \mathbb{R}^{I} \to \mathbb{R}^{n}, \quad \pi(\boldsymbol{\mu}) = (\mu_1(1), \dots, \mu_n(1)), \tag{21}$$

is the coordinate-erasing projection that erases all pseudomarginals not representing the input variables (see Step 1 of the algorithm).

Fig. 4 shows the constructed reparameterized Potts problem for an example input polyhedron.

As for each $a_{ij} \neq 0$ a constant number of objects and object pairs is created, the encoding time is $O(\mathrm{size}(\mathbf{A}))$.

5 Obtained Reductions

In Sections 3 and 4 we described our core construction. Here we describe several reductions that are more or less straightforward consequences of this construction.

5.1 Reduction from LF and LP

The *linear feasibility problem* is the problem of solving a system of linear inequalities. In our formulation, given a matrix $\mathbf{A}$ (with rational entries) the aim is to decide if the polyhedron P is nonempty and if so, to find an element $\mathbf{x} \in P$.

Theorem 2. *The linear feasibility problem reduces in linear time to the LP relaxation of the Potts problem with three labels.*

Proof. If P is bounded, the claim holds by composing the reductions in Sections 3 and 4. If P is unbounded, it has at least one vertex. The reduction in Section 4 cuts off a part of P because the pseudomarginals are bounded by 1. But, due to the scaling in Section 3.3, this part does not contain any vertex. Therefore, cutting this part off preserves at least some solutions to the input problem. □

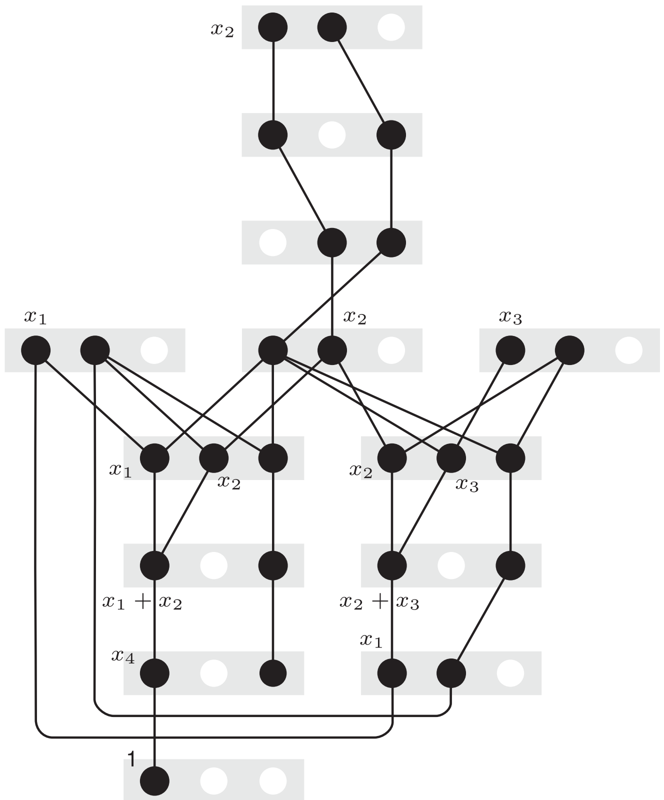

Fig. 4. A reparameterized Potts problem that encodes the polyhedron $P = \{\, (x_1, x_2, x_3, x_4) \in \mathbb{R}^4 \,|\, x_1 + x_2 = x_4,\ x_2 + x_3 = x_1,\ x_1 \geq 0,\ x_2 \geq 0,\ x_3 \geq 0,\ x_4 = 1 \,\}$.

The *linear programming problem* is the problem of minimizing a linear function subject to linear inequalities.

Lemma 3. *The linear programming problem reduces in linear time to the linear feasibility problem.*

Proof. By strong duality, any linear program

$$\min\{\, \langle \mathbf{c}, \mathbf{x} \rangle \,|\, \mathbf{A}\mathbf{x} = \mathbf{b},\ \mathbf{x} \geq \mathbf{0} \,\},$$

can be solved by solving the system

$$\mathbf{A}\mathbf{x} = \mathbf{b},\ \ \mathbf{A}^T\mathbf{y} + \mathbf{z} = \mathbf{c},\ \ \langle \mathbf{c}, \mathbf{x} \rangle = \langle \mathbf{b}, \mathbf{y} \rangle,\ \ \mathbf{x}, \mathbf{z} \geq \mathbf{0}.$$

Plugging $\mathbf{y} = \mathbf{y}_+ - \mathbf{y}_-$ where $\mathbf{y}_+, \mathbf{y}_- \geq \mathbf{0}$ puts this system into form 9. The system is feasible if and only if the input LP is feasible and bounded. The reduction takes linear time because it essentially copies $\mathbf{A}, \mathbf{b}, \mathbf{c}$ twice to the output. □

This gives us the central result of our paper.

Theorem 4. *The linear programming problem reduces in linear time to the LP relaxation of the Potts problem with three labels.*

Proof. Combine Lemma 3 and Theorem 2. □

5.2 Polyhedral Interpretation

Composing the reductions done in Sections 3 and 4 (see equality (20)) yields that the input polytope P is a (scaled) coordinate-erasing projection of a face of a local marginal polytope Λ. This recovers our result [9, Theorem 1] with the constraint that the output problem is a (reparameterized) Potts problem. Here we reformulate this result in terms of the relaxed Potts polytope Π.

By moving the numbers $\varphi_{uv}(k)$ to the unary costs, (20) can be expressed in terms of Π rather than Λ. Defining vector $\mathbf{h} \in \mathbb{R}^{(V \times K) \cup E}$ by $h_u(k) = g_u(k) - \sum_{v \in N_u} \varphi_{uv}(k)$ and $h_{uv} = 2$, we indeed have

$$\pi\Big(\arg\min_{\boldsymbol{\mu} \in \Lambda} \langle \mathbf{g}, \boldsymbol{\mu} \rangle\Big) = \pi'\Big(\arg\min_{\nu \in \Pi} \langle \mathbf{h}, \nu \rangle\Big), \tag{22}$$

where $\pi'\colon \mathbb{R}^{(V \times K) \cup E} \to \mathbb{R}^n$ is the coordinate-erasing projection given by $\pi'(\nu) = (\nu_1(1), \ldots, \nu_n(1))$. Comparing (20) with (22) yields the following result.

Theorem 5. *Every polytope is (up to scale) a coordinate-erasing projection of a face of a relaxed Potts polytope with three labels, whose description (by a set of linear inequalities) can be computed from the description of the input polytope in linear time.*

5.3 Relation to the Dual LP Relaxation

Rather than linear program (2) it is often better to solve its dual. As shown, e.g., in [6], the dual LP relaxation maximizes the function

$$L(\mathbf{g}) = \sum_{u \in V} \min_{k \in K} g_u(k) + \sum_{\{u,v\} \in E} \min_{k,\ell \in K} g_{uv}(k, \ell), \tag{23}$$

over reparameterization of $\mathbf{g}$ (i.e., we maximize $L(\mathbf{g}')$ over φ, where $\mathbf{g}'$ is given by (4). Function (23) is a lower bound on (2),

$$\min_{\boldsymbol{\mu} \in \Lambda} \langle \mathbf{g}, \boldsymbol{\mu} \rangle \geq L(\mathbf{g}). \tag{24}$$

By strong duality, inequality (24) holds with equality if and only if $\mathbf{g}$ is dual-optimal, i.e., no reparameterization of $\mathbf{g}$ can increase the lower bound.

For the Potts problem, the dual optimal value does not change if we add to the dual the constraints

$$\varphi_{uv}(k) + \varphi_{vu}(k) = 0, \qquad \{u, v\} \in E,\ k \in K \tag{25a}$$

$$|\varphi_{uv}(k)| \leq \frac{1}{2} h_{uv}, \quad u \in V,\ v \in N_u,\ k \in K. \tag{25b}$$

This is proved by writing the dual of the Kleinberg-Tardos relaxation (7), see Theorem 10 in Appendix, which can be found on the Computer Society Digital Library at http://doi.ieeecomputersociety.org/10.1109/TPAMI.2016.2582165. Note that the numbers $\varphi_{uv}(k)$ used in our gadgets satisfy (25).

Let us emphasize that the reduction from Section 4 applies only to the primal LP relaxation. The question whether there is a linear-time reduction of the general LP to the dual LP relaxation is left open in this paper. Does our reduction relate in any way to the dual LP relaxation? The reparameterized Potts problem constructed in Section 4.2 satisfies (17), hence it has $L(\mathbf{g}) = 0$. Therefore:

- If $P = \emptyset$ then $\min_{\boldsymbol{\mu} \in \Lambda} \langle \mathbf{g}, \boldsymbol{\mu} \rangle > L(\mathbf{g}) = 0$.
- If $P \neq \emptyset$ then $\min_{\boldsymbol{\mu} \in \Lambda} \langle \mathbf{g}, \boldsymbol{\mu} \rangle = L(\mathbf{g}) = 0$.

This shows that the linear feasibility problem in fact reduces to a simpler problem than the LP relaxation (2), namely, to deciding whether $\mathbf{g}$ is dual optimal.

Theorem 6. *The linear feasibility problem reduces in linear time to the following problem: given a reparameterized Potts problem with three labels, decide if its cost vector is optimal to the dual LP relaxation.*

5.4 Reduction to Multiway Cut Problem

Closely related to the Potts problem is the *multiway cut problem*. For its LP relaxation, given in [17], we prove a result analogous to Theorem 4.

A multiway cut in a graph $(V \cup K, E')$, where K are terminals and $E' \subseteq \binom{V \cup K}{2}$, is a subset of edges whose removal leaves each terminal in a separate component. Given edge costs $h'_{uv} \geq 0$, the goal of the multiway cut problem is to find a multicut with minimum total cost. The LP relaxation of this problem [17] reads

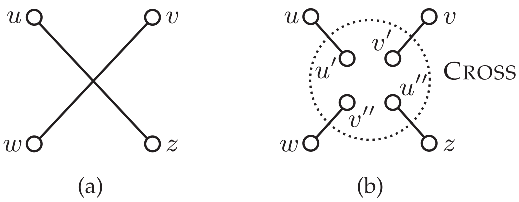

Fig. 5. Eliminating an edge crossing.

$$\text{minimize} \sum_{\{u,v\}\in E'} \frac{1}{2} h'_{uv} \sum_{k\in K} |\mu_u(k) - \mu_v(k)| \tag{26a}$$

$$\text{subject to} \sum_{k\in K} \mu_u(k) = 1, \quad u \in V \cup K \tag{26b}$$

$$\mu_k(k) = 1, \qquad k \in K \tag{26c}$$

$$\mu_u(k) \geq 0, \qquad u \in V,\ k \in K. \tag{26d}$$

Theorem 7. *The LP relaxation of the Potts problem reduces in linear time to the LP relaxation of the multiway cut problem.*

Proof. As shown, e.g., in [12, Section 7.1], the Potts problem with graph (V,E), unary costs $g_u(k)$, and Potts costs h_{uv} reduces to the multiway cut problem with graph $(V\cup K, E')$ where $E' = E \cup \{\,\{u,k\} \mid u \in V,\ k \in K\,\}$, and costs $h'_{uv} = h_{uv}$ for $\{u,v\} \in E$ and $h'_{uk} = c_u - g_u(k)$ for $u \in V$, $k \in K$, where $c_u = \max_k g_u(k)$. Since $|K|$ is constant in our case, the reduction takes linear time.

We show that this reduction preserves the LP relaxation. The objective (26a) reads

$$\begin{aligned}&\sum_{\{u,v\}\in E} \frac{1}{2} h_{uv} \sum_{k\in K} |\mu_u(k) - \mu_v(k)| \\ &+ \sum_{u\in V}\sum_{k\in K} \frac{1}{2} [c_u - g_u(k)] \sum_{\ell\in K} |\mu_u(\ell) - \mu_k(\ell)|.\end{aligned} \tag{27}$$

Using (26c) we have

$$\sum_{\ell\in K} |\mu_u(\ell) - \mu_k(\ell)| = [1-\mu_u(k)] + \sum_{\ell\neq k} \mu_u(\ell) = 2[1-\mu_u(k)],$$

so the second sum in 27 is

$$\sum_{u\in V}\sum_{k\in K} [c_u - g_u(k)][1-\mu_u(k)] = C + \sum_{u\in V}\sum_{k\in K} g_u(k)\mu_u(k).$$

Therefore (27) equals (6) up to a constant C. □

Theorem 8. *The linear programming problem reduces in linear time to the LP relaxation of the multiway cut problem with three terminals.*

Proof. Combine Theorems 4 and 7. □

5.5 Reduction to Planar Potts Problem

As in [9], reduction to the Potts problem is possible even if this problem is required to have planar structure, at the expense of increasing the reduction complexity.

Theorem 9. *The linear programming problem reduces in quadratic time to the LP relaxation of the planar Potts problem with three labels, whose size is quadratic.*

Proof. Consider the reparameterized Potts problem constructed in Section 4.2, with graph (V,E). We will replace this problem with a planar reparameterized Potts problem with the same LP relaxation.

Let the graph (V,E) be drawn in the plane, such that the vertices are distinct points and the edges are line segments connecting the vertices. We assume w.l.o.g. that no three edges intersect at a common point, except at graph vertices. We will replace every edge crossing with a planar reparameterized Potts problem.

Let $\{u,z\}, \{v,w\} \in E$ be a pair of crossing edges, as shown in Fig. 5a. This pair of edges is replaced by a construction outlined in Fig. 5b. Object u is linked to object u' and v' is linked to v' using COPY. Object z is linked to u'' and w is linked to v'', setting $g_{u''z} = g_{uz}$ and $g_{v''w} = g_{vw}$. The encircled objects are linked to a gadget, named CROSS, that enforces unary pseudomarginals in objects u', u'' and v', v'' to be equal. If necessary, labels are again permuted using PERMUTE. The construction can be drawn arbitrarily small so that it is not intersected by any other edges.

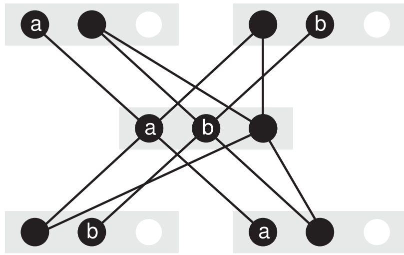

Fig. 6. Gadget CROSS.

The CROSS gadget is shown in Fig. 6. It is composed of four ADD1 gadgets. It works correctly only if $a + b \leq 1$. To ensure this, all pseudomarginals representing input variables in the output problem are scaled down by the factor of 2. This can be done by replacing the equation $x_n = 1$ in (9) with $x_n = \frac{1}{2}$, where the constant $\frac{1}{2}$ is constructed similarly as in Section 3.2, which can be done using a reparameterized Potts problem with planar structure.

Since the total number of edge crossings in a graph (V,E) is $O(|E|^2)$, the reduction time and the output size are quadratic. □

The encoding time and the size of the output can be improved using [9, Lemma 10].

6 CONCLUDING REMARKS

We have constructed a linear-time reduction from the general linear program to the LP relaxation of the Potts problem with three labels. This shows that there is little hope to find a very efficient algorithm (based, e.g., on simple combinatorial principles) to solve the LP relaxation of the Potts problem. This negative result applies also to labeling problems with metric and semimetric pairwise cost functions (of which Potts is a special case), which often arise in computer vision [1], [2].

Let us compare this result with our previous work [9] where we constructed a linear-time reduction from LP to the LP relaxation of the min-sum problem with costs in $\mathbb{Z} \cup \{\infty\}$ [9, Theorem 2] and a quadratic-time reduction from LP to the LP relaxation of the min-sum problem with costs in $\mathbb{Z}$ [9, Theorem 9].

Theorem 4 is stronger than [9, Theorem 9] because the output min-sum problem (being the Potts problem) has costs in $\mathbb{Z}$ and our reduction is in linear time.

Theorem 4 is stronger than [9, Theorem 2] because there costs $\mathbb{Z} \cup \{\infty\}$ are needed for linear-time reduction. However, there is a price for this. The reduction [9, Theorem 2] has the desirable property of preserving approximation ratio: if a sub-optimal (i.e., feasible) solution of the LP relaxation of the output min-sum problem is found, the ratio of the optimal and suboptimal objective value is the same as for the input LP. We did not mention this property in [9] but it is rather obvious. Reductions with finite output costs do not have this property. It is open whether there exists a linear-time reduction from the general LP to the LP relaxation of the min-sum problem with finite costs (or even the Potts problem) that preserves approximation ratio.

Another difference from [9] is that there we encoded the input polyhedron directly by a min-sum problem while here we first preprocess it to the form with coefficients $\{-1, 0, 1\}$. In fact, this preprocessing could be used to simplify the reduction in [9].

On the other hand, our results in this paper could be proved in an alternative way, shorter but less transparent. In [9, Section 4.2] we constructed a linear-time reduction from LF to the LP relaxation (2) of the min-sum problem with costs in $\{0, \infty\}$. This LP relaxation has the form 9 with coefficients $\{-1, 0, 1\}$ and every $\mathbf{x} \in P$ satisfying $\mathbf{A}^{+}\mathbf{x} = \mathbf{A}^{-}\mathbf{x} \leq \mathbf{1}$, so it can be encoded by a Potts problem as described in Section 4.

Finally, our work is related to [18], [19] where polyhedral universality results similar to our Theorem 5 are derived for the three-way transportation polytope and the traveling salesman polytope. However, the reduction time in these works is not shown to be linear.

Acknowledgments

This work has been supported by the Czech ministry of education, youth and sports under the ERC-CZ grant LL1303.

References

[1] R. Szeliski, et al., "A comparative study of energy minimization methods for Markov random fields with smoothness-based priors," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 6, pp. 1068–1080, Jun. 2008.

[2] J. H. Kappes, et al., "A comparative study of modern inference techniques for structured discrete energy minimization problems," *Int. J. Comput. Vis.*, vol. 115, no. 2, pp. 155–184, 2015.

[3] S. Živný, *The Complexity of Valued Constraint Satisfaction Problems*. Berlin, Germany: Springer, 2012.

[4] M. J. Wainwright and M. I. Jordan, "Graphical models, exponential families, and variational inference," *Found. Trends Mach. Learn.*, vol. 1, no. 1–2, pp. 1–305, 2008.

[5] M. I. Shlezinger, "Syntactic analysis of two-dimensional visual signals in noisy conditions," *Cybern. Syst. Anal.*, vol. 12, no. 4, pp. 612–628, 1976.

[6] T. Werner, "A linear programming approach to max-sum problem: A review," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 29, no. 7, pp. 1165–1179, Jul. 2007.

[7] A. Koster, S. P. van Hoesel, and A. W. Kolen, "The partial constraint satisfaction problem: Facets and lifting theorems," *Oper. Res. Lett.*, vol. 23, no. 3–5, pp. 89–97, 1998.

[8] V. Kolmogorov, J. Thapper, and S. Živný, "The power of linear programming for general-valued CSPs," *SIAM J. Comput.*, vol. 44, no. 1, pp. 1–36, 2015.

[9] D. Průša and T. Werner, "Universality of the local marginal polytope," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 37, no. 4, pp. 898–904, Apr. 2015.

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

[11] C. Rother, V. Kolmogorov, V. S. Lempitsky, and M. Szummer, "Optimizing binary MRFs via extended roof duality," in *Proc. Conf. Comput. Vis. Pattern Recognit.*, 2007, pp. 1–8.

[12] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 23, no. 11, pp. 1222–1239, Nov. 2001.

[13] J. Kleinberg and E. Tardos, "Approximation algorithms for classification problems with pairwise relationships: Metric labeling and Markov random fields," *J. ACM*, vol. 49, no. 5, pp. 616–639, 2002.

[14] C. Chekuri, S. Khanna, J. Naor, and L. Zosin, "A linear programming formulation and approximation algorithms for the metric labeling problem," *SIAM J. Discrete Math.*, vol. 18, no. 3, pp. 608–625, 2005.

[15] J. Chuzhoy and J. Naor, "The hardness of metric labeling," *SIAM J. Comput.*, vol. 36, no. 5, pp. 1376–1386, 2007.

[16] P. Kumar, "Rounding-based moves for metric labeling," in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 109–117.

[17] G. Călinescu, H. Karloff, and Y. Rabani, "An improved approximation algorithm for multiway cut," in *Proc. 30th Annu. ACM Symp. Theory Comput.*, 1998, pp. 48–52. [Online]. Available: http://doi.acm.org/10.1145/276698.276711

[18] J. A. De Loera and S. Onn, "All linear and integer programs are slim 3-way transportation programs," *SIAM J. Optim.*, vol. 17, no. 3, pp. 806–821, 2006.

[19] L. J. Billera and A. Sarangarajan, "All 0-1 polytopes are traveling salesman polytopes," *Combinatorica*, vol. 16, no. 2, pp. 175–188, 1996.