SIAM J. OPTIM.
Vol. 29, No. 3, pp. 1745–1771

SOLVING LP RELAXATIONS OF SOME NP-HARD PROBLEMS IS AS HARD AS SOLVING ANY LINEAR PROGRAM*

DANIEL PRŮŠA† AND TOMÁŠ WERNER†

Abstract. We show that the general linear programming (LP) problem reduces in nearly linear time to the LP relaxations of many classical NP-hard combinatorial problems, assuming sparse encoding of instances. We distinguish two types of such reductions. In the first type (shown for set cover/packing, facility location, maximum satisfiability, maximum independent set, and multiway cut), the input linear program is feasible and bounded iff the optimum value of the LP relaxation attains a threshold, and then optimal solutions to the input linear program correspond to optimal solutions to the LP relaxation. In the second type (shown for exact set cover, three-dimensional matching, and constraint satisfaction), feasible solutions to the input linear program correspond to feasible solutions to the LP relaxations. Thus, the reduction preserves objective values of all (not only optimal) solutions. In polyhedral terms, every polytope in standard form is a scaled coordinate projection of the optimal or feasible set of the LP relaxation. Besides nearly linear-time reductions, we show that the considered LP relaxations are P-complete under log-space reductions, and therefore also hard to parallelize. These results pose a limitation on designing algorithms to compute exact or even approximate solutions to the LP relaxations, as any lower bound on the complexity of solving the general LP problem is inherited by the LP relaxations.

Key words. linear programming relaxation, combinatorial optimization, nearly linear-time reduction, log-space reduction, convex polytope, universality, LP-completeness, extension complexity

AMS subject classifications. 52B12, 68Q17, 90C05, 90C06, 90C27

DOI. 10.1137/17M1142922

1. Introduction.

NP-hard problems in combinatorial optimization can usually be expressed as 0-1 linear programs with natural linear programming (LP) relaxations. Solutions to these relaxations are useful for computing exact optimal solutions (by branch-and-bound methods using the LP relaxation to compute lower bounds), approximate solutions (by rounding schemes), or exact optimal solutions for tractable problem subclasses (that is, those with zero integrality gap). Although LP relaxations can be solved in polynomial time [18], solving them for large instances can be inefficient or impossible. Applications leading to large-scale combinatorial optimization nowadays frequently appear in disciplines dealing with "big data", such as computer vision, machine learning, artificial intelligence, data mining, or data science.

It is therefore natural to ask if some LP relaxations are easier to solve than others, that is, if the structures of some problems allow us to design algorithms that would solve the LP relaxations more efficiently than general LP algorithms. An example is the LP relaxations that can be reduced in linear time to the max-flow problem. This class includes linear programs with up to two nonzeros per column [15], which in addition have half-integral solutions. Such a reduction is important in practice. For example, it is a core tool for solving discrete energy minimization tasks arising

*Received by the editors August 10, 2017; accepted for publication (in revised form) April 22, 2019; published electronically July 2, 2019. A preliminary version of this article appeared in the Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms, 2017, pp. 1372–1382.

https://doi.org/10.1137/17M1142922

Funding: The second author was supported by the Czech Operational Programme "Research, Development and Education" project CZ.02.1.01/0.0/0.0/16 019/0000765.

†Faculty of Electrical Engineering, Czech Technical University, Karlovo nam. 13, 121 35, Prague 2, Czech Republic (prusa@fel.cvut.cz, werner@fel.cvut.cz).

in computer vision [17], which is enabled by the availability of very fast max-flow solvers [2, 13] that handle large sparse input graphs easily (in linear space and empirically almost linear time). Another example is positive linear programs (PLPs): there are algorithms that compute approximate solutions to PLPs much faster than the general LP algorithms compute exact solutions, and such approximations often allow good approximations to the original combinatorial problem [22, 34] to be constructed.

We show that LP relaxations of many classical NP-hard combinatorial optimization problems are not easier to solve than general linear programs. Precisely, we show the following.

1. Searching for a nonnegative solution of a system of linear inequalities (the linear feasibility problem in equality form, LFE) can be reduced to each of the LP relaxations in nearly linear time, assuming the random access machine (RAM) model of computation and sparse encoding of instances. More precisely, deciding the feasibility of LFE reduces to deciding if the optimum value of the LP relaxation attains a threshold, and if so then every optimal solution to the LP relaxation is, after omitting auxiliary variables and scaling, a solution to LFE. As the LFE problem is linear-time equivalent to the LP problem, this implies that each LP relaxation is *LP-complete* under reductions in nearly linear time. In polyhedral terms, this means that every polytope in standard form is, up to scale, a coordinate projection of the set of optimal solutions to the LP relaxation, where this LP relaxation can be constructed from the polytope in nearly linear time.

2. For some of the LP relaxations we show a stronger result: feasible solutions to the LP relaxation are, after omitting auxiliary variables and scaling, in bijection with solutions to the LFE instance. That is, every polytope in standard form is a scaled coordinate projection of the feasible set of the LP relaxation. This implies a nearly linear-time reduction from the LP problem to each LP relaxation that preserves all objective values.

We construct these reductions in two steps. First (in section 3) we reduce the LFE problem in nearly linear time to its restricted form with 0-1 coefficients and at most three variables per equation (we call this LFE-BIN3). Then (in sections 4 and 5) we reduce this intermediate form in linear time to the LP relaxations.

3. Though our main focus is on time complexity, we also show (in section 6) that the LP relaxations considered are P-complete under log-space reductions.

In our previous work we showed the hardness of LP relaxation for two particular problems, the valued CSP [28] and uniform metric labeling [29]. Here we present a more general framework, which simplifies the proofs in [28, 29] and allows us to obtain the hardness result for more problems. A preliminary version of this article appeared as [27]. To the best of our knowledge, there is not much other literature on the hardness of LP relaxations, although works showing the universality of certain polytopes are related. Thus, [8] shows that every polytope is a projection of the slim three-dimensional (3-D) transportation polytope. In contrast to our strong reductions, this reduction is not in nearly linear time but does not use scaling. Integral hulls of some NP-hard combinatorial problems are universal, e.g., [1] shows that every polytope with 0-1 vertices is affinely equivalent to a face of some traveling salesman polytope. Unfortunately, this does not say much about the hardness of LP relaxations.

2. Overview. To start the exposition, we describe our setup and summarize the chains of reductions that lead to our main results. All the main results are proved here, up to the reduction from LFE to LFE-BIN3 and the reductions from LFE-BIN3

to the LP relaxations, which are proved later in sections 3, 4, and 5.

2.1. Assumptions. The complexity of reductions depends on the computational model and on the way of encoding instances. As our computational model, we adopt the RAM [26, section 2.6].

Instances of linear programming/feasibility problems are lists of rational numbers, i.e., rational vectors. One can consider two ways of storing such a vector $a = (a_1, \dots, a_n) \in \mathbb{Q}^n$ in the memory. In *array encoding*, we simply store the list $a_1, \dots, a_n$. In *index-value encoding*, we store the list of pairs (i, a_i) for all *nonzero* components of a. Clearly, the array encoding of a vector can be transformed to its index-value encoding in nearly linear time, but not *vice versa*. Recall that a function $f\colon \mathbb{N} \to \mathbb{N}$ is *nearly linear* [14, 25] if $f(n) = O(n \log^k n)$ for some $k \in \mathbb{N}$.

We assume that both the inputs and outputs of our reductions are in index-value encoding. However, as an array encoding can be transformed to an index-value one in nearly linear time, our main results (nearly linear-time reductions from LP problems to LP relaxations) would not change if the input were in the array encoding. In contrast, the assumption that the outputs (LP relaxations) are in index-value encoding cannot be dropped, since transforming them to array encoding would in general take more than nearly linear time. But note that array encoding is unnatural for (LP relaxations of) combinatorial decision/optimization problems. Index-value encoding corresponds to the usual encoding of instances of such problems, where, e.g., a (hyper)graph is represented by a list of edges, rather than by an adjacency matrix in array encoding.

The total number of bits needed to store the index-value encoding of a rational vector $a = (a_1, \dots, a_n) \in \mathbb{Q}^n$ is, up to multiplicative and additive constants,

$$\text{size}(a) = d\lceil \log_2(n+1) \rceil + L(a), \qquad \text{where} \qquad L(a) = \sum_{i=1}^{n} \lceil \log_2(|p_i q_i| + 1) \rceil. \tag{1}$$

Here, d is the number of nonzero components of a and $p_i \in \mathbb{Z}$ and $q_i \in \mathbb{N}$ are such that $a_i = p_i/q_i$, where q_i does not divide p_i unless $q_i = 1$ or $p_i = 0$. The first term in (1) accounts for indices, and the second term $L(a)$ for values. For a list $(A, b, \dots)$ of rational matrices and vectors, $\text{size}(A, b, \dots)$ denotes the size of the vector formed by all their entries.

2.2. Easy reductions. The *linear feasibility problem* (henceforth abbreviated as LF) aims to solve a system of linear inequalities, $Ax \le b$. The *linear feasibility problem in equality form* (LFE) aims to find a nonnegative solution to a system of linear equations, i.e., to solve the system $Ax = b$, $x \ge 0$. The *linear programming problem* (LP problem) aims to minimize a linear function $c^T x$ subject to $Ax \le b$. *Linear optimization over LFE* (abbreviated as LP/LFE) aims to minimize $c^T x$ subject to $Ax = b$ and $x \ge 0$. We assume that $A = [a_{ij}] \in \mathbb{Q}^{m \times n}$, $b = (b_1, \dots, b_m) \in \mathbb{Q}^m$, $c = (c_1, \dots, c_n) \in \mathbb{Q}^n$, and $x = (x_1, \dots, x_n) \in \mathbb{R}^n$.

THEOREM 2.1. *The LF problem reduces in linear time to the LFE problem.*

Proof. A system $Ax \le b$ is equivalent to the system $Ay - Az + u = b$, $y, z, u \ge 0$, which is an LFE instance. The solutions (y, z, u) to the LFE instance map surjectively to the solutions x to the LF instance via $x = y - z$. The reduction clearly takes linear time because its job is just to construct the matrix $\begin{bmatrix} A & -A & I \end{bmatrix}$ from matrix A. ∎

THEOREM 2.2. *The LP problem reduces in linear time to the LF problem.*

Proof. By LP duality, solving a linear program $\min\{c^T x \mid Ax \le b\}$ is equivalent to solving the system $Ax \le b$, $y \le 0$, $A^T y = c$, $c^T x = b^T y$, which is an LF instance. In

particular, the LP instance is feasible and bounded iff the LF instance is feasible. The solutions (x, y) to the LF instance are, after omitting the variables y, in bijection with the optimal solutions to the LP instance. Linear time would be shown in a similar way to that in the previous theorem. ∎

THEOREM 2.3. *There is a linear-time reduction from the LP problem to the LP/LFE problem that preserves objective values of feasible solutions.*

Proof. Minimizing $c^T x$ subject to $Ax \leq b$ is equivalent to minimizing $c^T(y - z)$ subject to $Ay - Az + u = b$, $y, z, u \geq 0$, which is an LP/LFE instance. The input linear program is feasible iff the output linear program is feasible. The objective value $c^T(y - z)$ of every feasible solution to the output linear program is trivially equal to the objective value $c^T x$ of the corresponding solution $x = y - z$ to the input linear program. ∎

2.3. Reduction to an intermediate form. The LFE problem can be reduced in nearly linear time to the LFE problem with 0-1 coefficients and at most three variables per equation, i.e., to the form $Ax = b$, $x \geq 0$, where the entries of A and b are in $\{0, 1\}$ and each row of A has at most three 1's. We call this restricted form LFE-BIN3. In Theorem 2.4, $\operatorname{ext} P$ denotes the set of extreme points (vertices) of a convex polyhedron P. A *coordinate projection*[1] is a projection that copies a subset of coordinates and deletes the remaining ones, that is, a map

$$\pi\colon \mathbb{R}^n \to \mathbb{R}^m, \quad \pi(x_1, \ldots, x_n) = (x_{\tau(1)}, \ldots, x_{\tau(m)}) \tag{2}$$

for some $m \leq n$ and some injection $\tau\colon \{1, \ldots, m\} \to \{1, \ldots, n\}$.

THEOREM 2.4. *There is an algorithm that, from any LFE instance, constructs in nearly linear time an LFE-BIN3 instance, a coordinate projection π, and a scale $\sigma \in \mathbb{Q}$, $0 < \sigma \leq 1$, such that*

$$\operatorname{ext} P \subseteq \pi(Q)/\sigma \subseteq P, \tag{3}$$

where P denotes the solution set of the LFE instance and Q denotes the solution set of the LFE-BIN3 instance.

Proof. See section 3. ∎

The coordinate projection π accounts for omitting the auxiliary variables introduced in the reduction. Why is the scale σ needed? While polyhedron P can be unbounded, Q is always bounded (since the variables and coefficients of LFE-BIN3 are nonnegative). Thus, the reduction in general cuts off a part of P. In the reduction, P is scaled down by σ so that none of its vertices is cut off. If P is bounded, (3) implies $P = \pi(Q)$. In general, (3) implies that $P = \emptyset$ iff $Q = \emptyset$, i.e., the LFE instance is feasible iff the LFE-BIN3 instance is feasible.

Let LP/LFE-BIN3 denote the linear optimization problem over LFE-BIN3.

COROLLARY 2.5. *There is a reduction in nearly linear time from the LP/LFE problem to the LP/LFE-BIN3 problem with these properties: the LP/LFE instance is feasible and bounded iff the LP/LFE-BIN3 instance is feasible, and in that case the objective values of all their feasible solutions are equal.*

Proof. We first describe the reduction for the case when the LP/LFE instance is feasible and bounded. Apply the reduction from Theorem 2.4 to the LFE problem. As

[1] Called *coordinate-erasing projection* in [8].

$P \neq \emptyset$, we have $\min\{c^T x \mid x \in P\} = \min\{c^T x \mid x \in \operatorname{ext} P\}$ because every nonempty polyhedron in the form $\{x \mid Ax = b,\ x \geq 0\}$ has a vertex. Therefore, (3) implies

$$\min\{c^T x \mid x \in P\} = \min\{c^T x \mid x \in \pi(Q)/\sigma\} = \min\{\sigma\pi^*(c)^T y \mid y \in Q\}, \tag{4}$$

where π^* denotes the adjoint of π. Thus, we have transformed the LP/LFE instance to an LP/LFE-BIN3 instance. Note that $\pi^*(c)$ is the vector c padded with zeros in the coordinates deleted by π. Since for every $y \in Q$ and $x = \pi(y)$ we trivially have $c^T x = \sigma\pi^*(c)^T y$, this reduction preserves the objective values of feasible solutions.

Note that the case when the LP/LFE instance is infeasible is also handled by reduction (4), because, by Theorem 2.4, we have $P = \emptyset$ iff $Q = \emptyset$.

This reduction can be generalized to the case when the input LP/LFE problem is feasible and unbounded.[2] By LP duality, a linear program $\min\{c^T x \mid Ax = b,\ x \geq 0\}$ is feasible and bounded iff the linear program

$$\min\{c^T x \mid Ax = b,\ x \geq 0,\ Ax' = b,\ x' \geq 0,\ A^T y' \leq c,\ c^T x' = b^T y'\} \tag{5}$$

is feasible, and they have the same optimal values. Thus, the above reduction applied to (5) has the desired properties. Since (5) is an instance of LP (rather than LP/LFE), we need to use the composition of the reductions from Theorems 2.1 and 2.4. ∎

2.4. Reduction to LP relaxations. The LFE-BIN3 problem can be reduced in linear time to the LP relaxations of a number of combinatorial optimization problems. We distinguish two types of such reductions.

The first type reduces LFE-BIN3 to the LP relaxation of the decision version (obtained by thresholding) of a combinatorial optimization problem. We present it for *minimum cost set cover/packing and their versions with unit costs, uncapacitated facility location, maximum satisfiability, maximum independent set (clique relaxation) and its unweighted version, and minimum multiway cut*.

THEOREM 2.6. *For each of the above LP relaxations, there is a linear-time algorithm that, for any instance of LFE-BIN3, constructs an instance of the LP relaxation, a threshold $t \in \mathbb{Q}$, and a coordinate projection π such that*

$$c^T x \geq t \quad \forall x \in Q, \tag{6a}$$

$$P = \pi(\{x \in Q \mid c^T x = t\}), \tag{6b}$$

where P denotes the solution set of the LFE-BIN3 instance, and Q and $c^T x$ denote the feasible set and objective function, respectively, of the LP relaxation instance.

Proof. See section 4. ∎

Condition (6a) says that t is a lower bound on the minimum value of the LP relaxation instance. By (6b), the LFE-BIN3 instance is feasible iff this bound is attained. In that case, the optimal solutions to the LP relaxation instance are, after omitting auxiliary variables, in bijection with the solutions to the LFE-BIN3 instance.

The second (stronger) type reduces LFE-BIN3 to the LP relaxation of a combinatorial *decision* problem. Thus, this LP relaxation is a linear feasibility problem. Endowing it with a linear objective function yields the LP relaxation of the optimization version of the problem. We present this reduction for *exact set cover, exact 3-D*

[2] This case could be more easily decided by solving another LFE-BIN3 instance (a Turing reduction). Yet we want to stick to reductions that call a solver to the output problem only once (which is analogous to many-one reductions for decision problems).

matching, and constraint satisfaction and their optimization versions *minimum-cost exact set cover,* 3*-D assignment, and valued constraint satisfaction*.

THEOREM 2.7. *For each of the above LP relaxations, there is a linear time algorithm that, for any instance of LFE-BIN*3*, constructs an instance of the LP relaxation and a coordinate projection* π *such that*

$$P = \pi(Q), \tag{7}$$

where P *denotes the solution set of the LFE-BIN*3 *instance and* Q *denotes the feasible set of the LP relaxation instance.*

Proof. See section 5. ∎

Equality (7) says that the feasible solutions to the LP relaxation are, after omitting auxiliary variables, in bijection with the solutions to the LFE-BIN3 instance. In particular, the LFE-BIN3 instance is feasible iff the LP relaxation is feasible.

2.5. Main results. We now combine the reductions stated so far with our main hardness results.

COROLLARY 2.8. *The LP problem reduces in nearly linear time to each LP relaxation in section* 4*.*

Proof. Compose the reductions from Theorems 2.2, 2.1, 2.4, and 2.6. ∎

More precisely, this reduction is such that the input LP instance is feasible and bounded iff the optimal value of the LP relaxation instance attains a threshold, and if so, then their optimal solutions are related by a composition of the affine map $x = y - z$ (see Theorem 2.1), coordinate projections, and a scaling.

COROLLARY 2.9. *There is a reduction in nearly linear time from the LP problem to each LP relaxation in section* 5 *that preserves the objective values of feasible solutions.*

Proof. Compose the reductions from Theorem 2.3 and Corollary 2.5, where in the proof of Corollary 2.5 we additionally apply the reduction from Theorem 2.7 to the feasibility set of LP/LFE-BIN3. ∎

This implies that solving any LP relaxation from section 5 only *approximately* (i.e., finding a feasible solution with the objective value near to the optimal value) is not easier than solving the LP problem approximately, with the same approximation factor.

These hardness results can also be formulated in terms of input and output polyhedra. We say that a convex polyhedron is in *equality* (or *standard form* if it is the solution set of an LFE instance). As usual, a *polytope* is a bounded convex polyhedron.

COROLLARY 2.10. *Every polytope in standard form is, up to scale, a coordinate projection of the set of optimal solutions to an instance of each LP relaxation from section* 4*. This LP relaxation can be constructed from the polytope (described by linear inequalities) in nearly linear time.*

Proof. Apply the reduction from Theorem 2.4 to the LFE representing the polytope. Since the polytope is bounded, the second inclusion in (3) holds with the equality. Now, apply the reduction from Theorem 2.6 to the resulting LFE-BIN3. Equality (6b) says that the polytope represented by this LFE-BIN3 is a coordinate projection of the optimal set of each LP relaxation. ∎

COROLLARY 2.11. *Every polytope in standard form is, up to scale, a coordinate projection of the feasible set of an instance of each LP relaxation from section* 5. *This LP relaxation can be constructed from the polytope (described by linear inequalities) in nearly linear time.*

Proof. The proof is as for Corollary 2.10, but we use the reduction from Theorem 2.7. ∎

Similar corollaries would also hold for the polytope in general form (i.e., the solution set of a bounded LF problem), but the coordinate projection would have to be precomposed with the affine map $x = y - z$ (see Theorem 2.1).

3. Reduction to an intermediate form. Here we show that the LFE problem can be reduced in nearly linear time to the LFE-BIN3 problem, thereby proving Theorem 2.4. We obtain this reduction in four steps, reducing the input LFE instance to an increasingly restricted form.

To prove the time complexity of these reductions, we will only show that the *size* of the output is (nearly) linear in the size of the input, and then it becomes obvious enough that the *time* of the reduction is also nearly linear. We assume, without loss of generality, that the matrix A of the input LFE instance has no zero column or row. We use $\bar{A} = [\bar{a}_{ij}] = \begin{bmatrix} A & b \end{bmatrix} \in \mathbb{Q}^{m\times(n+1)}$ to denote the extended matrix of the system $Ax = b$.

THEOREM 3.1. *The LFE problem reduces in linear time to the LFE problem with integer coefficients.*

Proof. For each rational nonzero input coefficient $a_{ij} = p_{ij}/q_{ij}$ with $p_{ij} \in \mathbb{Z}$ and $q_{ij} \in \mathbb{N}$, we introduce an auxiliary variable y_{ij} and the equation $q_{ij}y_{ij} = |p_{ij}|x_j$. Then, in the input system we replace each nonzero term $a_{ij}x_j$ with $\operatorname{sgn}(a_{ij})y_{ij}$. Coefficients b_i are handled similarly. This clearly takes linear time.[3] ∎

Example 3.2. The system

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

is transformed to the system

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

THEOREM 3.3. *The LFE problem with integer coefficients reduces in nearly linear time to the LFE problem with coefficients* $\{-1, 0, 1\}$.

Proof. The idea is similar to [8, section 3.1]. Suppose we want to construct the product ax for a coefficient $a \in \mathbb{N}$ and a variable x. We create the system

$$\begin{aligned} x_1 &= x_0 + y_0, & y_0 &= x_0 = x, \\ x_2 &= x_1 + y_1, & y_1 &= x_1, \\ &\vdots & &\vdots \\ x_d &= x_{d-1} + y_{d-1}, & y_{d-1} &= x_{d-1}. \end{aligned} \tag{8}$$

[3] Note that the most obvious reduction, multiplying all coefficients of each equation by the least common multiple of their denominators, needs superlinear time.

The first line of the system enforces $x_1 = 2x_0$, the second line enforces $x_2 = 2x_1$, etc. In general, $x_k = 2^k x$. The product ax can be now obtained by summing appropriate bits of the binary code for a, e.g., $11x_0 = x_0 + x_1 + x_3$ because $11 = 2^0 + 2^1 + 2^3$.

Given an input LFE instance $Ax = b$, $x \geq 0$ with $\begin{bmatrix} A & b \end{bmatrix} = \bar{A} \in \mathbb{Z}^{m \times (n+1)}$, the reduction proceeds as follows:

1. for each $j = 1, \ldots, n+1$, create system (8) with $d = \lfloor \log_2 \max_{i=1}^{m} |\bar{a}_{ij}| \rfloor$;
2. for each $i = 1, \ldots, m$, construct nonzero terms $a_{ij}x_j$ and b_i and compose the ith equation of the input system $Ax = b$ from them.

System (8) created in step 1 for one j has $O(\log_2 \max_i |\bar{a}_{ij}|)$ equations, each containing at most three nonzero coefficients. The total number of equations as well as variables created in step 1 is $O(L(\bar{A}))$ because

$$\sum_j \log_2 \max_i |\bar{a}_{ij}| \leq \sum_{i,j} \log_2(|\bar{a}_{ij}| + 1) \leq L(\bar{A}).$$

In step 2, the total number of terms representing one term $a_{ij}x_j$ and b_i is $O(L(a_{ij}))$ and $O(L(b_i))$, respectively. Hence, the system produced by step 2 has $O(L(\bar{A}))$ nonzero coefficients (and uses only the variables from step 1). To summarize, the index-value encoding of the output system has size

$$O(L(\bar{A}) \log(L(\bar{A}))) = O(\text{size}(\bar{A}) \log(\text{size}(\bar{A})))$$

and it is constructed in time linear in this size. ∎

Example 3.4. The system

$$\begin{aligned} 2x_1 + 11x_2 &= 1, \\ 3x_1 - 6x_2 &= 5 \end{aligned}$$

is transformed to the system

$$\begin{aligned} &x_{11} = x_{10} + y_{10}, \quad && y_{10} = x_{10} = x_1, \quad && x_{31} = x_{30} + y_{30}, \quad && y_{30} = x_{30} = 1, \\ &x_{21} = x_{20} + y_{20}, \quad && y_{20} = x_{20} = x_2, \quad && x_{32} = x_{31} + y_{31}, \quad && y_{31} = x_{31}, \\ &x_{22} = x_{21} + y_{21}, \quad && y_{21} = x_{21}, \quad && x_{11} + (x_{20} + x_{21} + x_{23}) = x_{30}, \\ &x_{23} = x_{22} + y_{22}, \quad && y_{22} = x_{22}, \quad && (x_{10} + x_{11}) - (x_{21} + x_{22}) = x_{30} + x_{32}. \end{aligned}$$

THEOREM 3.5. *The LFE problem with coefficients* $\{-1, 0, 1\}$ *reduces in linear time to the LFE problem involving only equations of the type* $x_i = 0$, $x_i = 1$, $x_i = x_j$, *and* $x_i + x_j = x_k$.

Proof. We show this with an example. Consider the equation $x_1 + x_2 - x_3 + x_4 = 1$. By moving negative terms to the other side, it can be rewritten as $x_1 + x_2 + x_4 = x_3 + 1$. This is replaced by $x_1 + x_2 = x_5$, $x_5 + x_4 = x_6$, $x_3 + x_7 = x_6$, $x_7 = 1$. If the initial equation had k variables, the number of auxiliary variables is $O(\log k)$. Thus, the reduction takes linear time. ∎

LEMMA 3.6. *For any matrix* $A = [a_{ij}] \in \mathbb{R}^{m \times m}$ *it holds that*

$$|\det A| \leq \prod_{j=1}^{m} \sum_{i=1}^{m} |a_{ij}|.$$

Proof. By Hadamard's inequality, $|\det A| \leq \prod_{j=1}^{m} \|a_j\|_2$, where $a_1, \ldots, a_m$ are the columns of A. Now we use that $\|a_j\|_2 \leq \|a_j\|_1 = \sum_{i=1}^{m} |a_{ij}|$. ∎

LEMMA 3.7. *Coordinates of every vertex $x = (x_1, \dots, x_n)$ of a convex polyhedron $\{x \in \mathbb{R}^n \mid Ax = b,\ x \geq 0\}$ with $\begin{bmatrix} A & b \end{bmatrix} = \bar{A} \in \mathbb{Z}^{m \times (n+1)}$ satisfy $x_j \leq 2^B$, where*[4]

$$B = \sum_{j=1}^{n+1} \left\lceil \log_2 \sum_{i=1}^{m} |\bar{a}_{ij}| \right\rceil. \tag{9}$$

Moreover, $B = O(L(\bar{A}))$, and expression (9) can be computed in time $O(\text{size}(\bar{A}))$.

Proof. It is well known from the theory of linear programming that the vector $x' = (x'_1, \dots, x'_p)$, $p \leq n$, of the nonzero coordinates of every vertex of the polyhedron is the solution to the system $A'x' = b'$, where A' is an invertible submatrix of A and b' is a subvector of b. By Cramer's rule, $x'_j = (\det A'_j)/(\det A')$, where A'_j denotes A' with the jth column replaced by b'. Since A' is invertible and has integer entries, $|\det A'| \geq 1$. By Lemma 3.6,

$$\left|\det A'_j\right| \leq \prod_{j=1}^{n+1} \sum_{i=1}^{m} |\bar{a}_{ij}| \leq 2^B.$$

Using bitwise arithmetic operations, expression (9) can be computed in linear time. To prove that $B = O(L(\bar{A}))$, write

$$B - n - 1 \leq \sum_{j=1}^{n+1} \log_2 \sum_{i=1}^{m} |\bar{a}_{ij}| \leq \sum_{j=1}^{n+1} \sum_{i=1}^{m} \log_2(|\bar{a}_{ij}| + 1) \leq L(\bar{A}),$$

where the second inequality follows from the fact that any nonnegative numbers $c_i = |\bar{a}_{ij}|$ satisfy $\sum_{i=1}^{m} c_i \leq \prod_{i=1}^{m}(c_i + 1)$. ∎

THEOREM 3.8. *The LFE problem with equations of the type $x_i = 0$, $x_i = 1$, $x_i = x_j$, and $x_i + x_j = x_k$ reduces in linear time to the LFE-BIN3 problem.*

Proof. We are facing a problem here: while the input LFE instance can be unbounded, the LFE-BIN3 instance is bounded (all its solutions have coordinates in the interval $[0, 1]$). Therefore, the reduction will in general cut off some part of the solution set of the input LFE. We make sure that this cutoff part contains no vertices.

To that end, the solution set of the input LFE is first scaled down so that all its vertices have coordinates in the interval $[0, 1]$. This is achieved by replacing each equation of the type $x_i = 1$ with the equation $x_i = \sigma$, where σ is the scale. Using Lemma 3.7, we need to set $\sigma = 2^{-B}$. This value of variable σ is obtained by repetitively halving the constant 1, by introducing the system

$$\begin{aligned} 1 = y_0 &= y_1 + z_1, & y_1 &= z_1, \\ y_1 &= y_2 + z_2, & y_2 &= z_2, \\ &\vdots & &\vdots \\ y_{B-1} &= y_B + z_B, & y_B &= z_B = \sigma. \end{aligned} \tag{10}$$

Note that these equations have the assumed types. By Lemma 3.7, the number of equations in (10) and the time needed to compute B are linear in the input size. Therefore, this step takes linear time.

[4] Bounds on vertex coordinates of a convex polyhedron with integer coefficients are well known; however, we do not know about any such bound with linear size, e.g., [26, Lemma 2.1] shows that $x_j \leq m!\,\alpha^{m-1}\beta$, where $\alpha = \max_{i,j} |a_{ij}|$ and $\beta = \max_i |b_i|$. The size of this bound is superlinear in $L(\bar{A})$, whereas the size of our bound 2^B is linear in $L(\bar{A})$.

Now we reduce the resulting LFE to LFE-BIN3. Equations of the types $x_i = 0$ and $x_i = 1$ already have the desired form. Observe that, for any $x_i, x_j \in [0,1]$, we have $x_i = x_j$ iff $x_i + x_k = 1$ and $x_j + x_k = 1$ for some $x_k \in [0,1]$. Therefore, each equation $x_i = x_j$ can be replaced with the system $x_i + x_k = 1$, $x_j + x_k = 1$. As this implicitly introduces the constraints $x_i, x_j \leq 1$, it cuts off some solutions (but no vertices). Similarly, each equation $x_i + x_j = x_k$ is replaced with the system $x_i + x_j + x_l = 1$, $x_k + x_l = 1$. ∎

The reductions from Theorems 3.1, 3.3, and 3.5 satisfy $P = \pi(Q)$, where P is the solution set of the input instance, Q is the solution set of the output instance, and π is the coordinate projection that deletes the auxiliary variables introduced in the reduction (such as the variables y_{ij} in Theorem 3.1). As the reduction from Theorem 3.8 cuts off some solutions but no vertices, it satisfies only (3). To conclude, composing the reductions from Theorems 3.1, 3.3, 3.5, and 3.8 yields Theorem 2.4.

For convenience, we further reduce the LFE-BIN3 problem by the following simple transformations. First, we make the right-hand sides of all equations equal 1, by replacing each equation $x_i = 0$ with $x_i + x_j = 1$ and $x_j = 1$. Now the system reads $Ax = 1$, $x \geq 0$, where $A \in \{0,1\}^{m \times n}$. This can equivalently[5] be written as

$$\sum_{i \in s} x_i = 1, \qquad s \in S, \tag{11a}$$

$$x_i \geq 0, \qquad i \in I, \tag{11b}$$

where I is a set of variables and $S \subseteq 2^I$ is a set of variable subsets such that $|s| \leq 3$ for all $s \in S$. Second, we achieve that $|s| \in \{1,3\}$[6] for all $s \in S$, by replacing each equation $x_i + x_j = 1$ with the equations $x_i + x_j + x_k = 1$, $x_k + x_l + x_p = 1$, $x_l = 1$.

We introduce some more notation related to (11), to be referred to in sections 4 and 5.

- $n_i = |\{s \in S \mid s \ni i\}|$ is the number of equations involving variable $i \in I$.
- $I_1 = \{i \in I \mid \{i\} \in S\}$ is the set of variables that occur in some single-variable equation.
- $S_3 = \{s \in S \mid |s| = 3\}$ is the set of three-variable equations. We assume that $\bigcup S_3 = I$, which can be achieved by introducing an equation $x_i + x_j + x_k = 1$ for every equation $x_i = 1$, where x_j, x_k are not in any other equation.
- $\phi\colon S_3 \times \{1,2,3\} \to I$ is a map such that $s = \{\phi(s,1), \phi(s,2), \phi(s,3)\}$ for all $s \in S_3$. This map introduces a local indexing of the variables in each three-variable equation. As $\bigcup S_3 = I$, this map is surjective.
- $\phi^{-1}(i) = \{(s,j) \in S_3 \times \{1,2,3\} \mid \phi(s,j) = i\}$ denotes the inverse of ϕ.

4. Weak reductions to LP relaxations. In this section we present weak reductions from the LFE-BIN3 problem to the LP relaxations, proving thus Theorem 2.6.

4.1. Set cover. Given a collection of n subsets of the set $\{1, \dots, m\}$, a *set cover* is a subset of the collection that covers all m elements. Given a nonnegative cost for each subset, the *minimum-cost set cover problem* seeks to find a set cover

[5] Assuming that no two rows of A are the same. This will indeed hold if no two equations of the input LFE instance are the same.

[6] Note that this final form cannot be further simplified in the following sense. If $|s| = 3$ for all $s \in S$, then system (11) is trivially satisfied by setting $x_i = \frac{1}{3}$ for all $i \in I$. If $|s| \in \{1,2\}$ for all $s \in S$ and system (11) is feasible, then it has a half-integral solution [15].

with minimum total cost. The well-known LP relaxation of this problem reads [36, Chapter 13]

$$\min\{c^T x \mid Ax \geq 1,\ x \geq 0\}, \tag{12}$$

where $A = [a_{ij}] \in \{0,1\}^{m \times n}$ is such that $a_{ij} = 1$ iff the jth subset contains element i, and $c \in \mathbb{Q}_+^n$ (where $\mathbb{Q}_+$ denotes the nonnegative rationals) are the costs.

THEOREM 4.1. *Let $c_j = \sum_{i=1}^m a_{ij}$ (that is, $c = A^T 1$). The optimal value of* (12) *is at least m, which is attained iff the variables x satisfy $Ax = 1$.*

Proof. For every feasible solution x to (12) we have $c^T x = 1^T Ax \geq 1^T 1 = m$. This inequality is tight iff $Ax = 1$. ∎

Theorem 4.1 implies Theorem 2.6, where $P = \{x \in \mathbb{R}^n \mid Ax = 1,\ x \geq 0\}$, $Q = \{x \in \mathbb{R}^n \mid Ax \geq 1,\ x \geq 0\}$, $t = m$, and π in this simple case is the identity map.

4.2. Set cover, unweighted. In the previous section, each subset in the set cover problem could have an arbitrary cost. Here we consider the restricted version with unit costs ($c_j = 1$ for all $j = 1, \ldots, n$), that is, the (unweighted) *minimum set cover problem*. We now show the reduction for this more difficult case.

Consider system (11) with $|s| \in \{1, 3\}$ for all $s \in S$. Consider the linear program

$$\min \quad \sum_{i \in I} (x_{i0} + x_{i1} + x_{i2}) \tag{13a}$$

$$\text{subject to} \quad \sum_{i \in s} x_{ij} \geq 1, \qquad s \in S_3,\ j \in \{0,1,2\}, \tag{13b}$$

$$x_{i0} \geq 1, \qquad i \in I_1, \tag{13c}$$

$$x_{i0} + x_{i1} + x_{i2} \geq 1, \qquad i \in I, \tag{13d}$$

$$x_{ij} \geq 0, \qquad i \in I,\ j \in \{0,1,2\} \tag{13e}$$

(note, S_3 and I_1 are defined in section 3). This linear program has the form (12) with $c = 1$.

THEOREM 4.2. *The optimal value of linear program* (13) *is at least $|I|$, which is attained iff variables x_{i0} ($i \in I$) satisfy* (11).

Proof. By (13d), the objective value of linear program (13) is at least $|I|$. This value is attained iff

$$x_{i0} + x_{i1} + x_{i2} = 1, \quad i \in I. \tag{14}$$

For $|s| = 1$, (13c) and (14) imply $x_{i0} = 1$, where $\{i\} = s$. For $|s| = 3$, (14) implies $\sum_{i \in s} x_{i0} + \sum_{i \in s} x_{i1} + \sum_{i \in s} x_{i2} = 3$. By (13b), each of the three terms is at least 1, and hence each term must be 1. In particular, $\sum_{i \in s} x_{i0} = 1$. Therefore, the values of x_{i0} form a feasible solution to (11).

For the other direction, let x_{i0} ($i \in I$) be a feasible solution to (11). For each $i \in I$, set $x_{i1} = x_{i2} = (1 - x_{i0})/2$. These values are feasible for (13) and satisfy (14). ∎

Theorem 4.2 implies Theorem 2.6, where P is the solution set of (11), Q is the feasible set of linear program (13), $t = |I|$, and π is the coordinate projection that copies the variables x_{i0} ($i \in I$) and deletes the auxiliary variables x_{i1}, x_{i2} ($i \in I$).

4.3. Set packing. Given n subsets of the set $\{1,\dots,m\}$, where each subset has a nonnegative weight, the *maximum-weight set packing problem* seeks to choose pairwise-disjoint subsets with maximum total weight. The well-known LP relaxation of this problem reads

$$\max\{c^T x \mid Ax \le 1,\ x \ge 0\}, \tag{15}$$

where A has the same meaning as in (12) and $c \in \mathbb{Q}_+^n$ are the weights. The reductions, for both the weighted and unweighted versions, are analogous to those for the set cover, up to the directions of some inequalities.

4.4. Facility location. Let F be a set of facilities, C be a set of cities, $E \subseteq F \times C$, $f\colon F \to \mathbb{Q}_+$ be the costs of opening facilities, and $c\colon E \to \mathbb{Q}_+$ be the costs of connecting cities to facilities. The *uncapacitated facility location problem* seeks to open a subset of facilities and, for each city, select one of the open facilities and assign the city to it such that the total cost is minimized. The problem has the well-known LP relaxation [36, Chapter 24]

$$\min \quad \sum_{(i,j)\in E} c_{ij}x_{ij} + \sum_{i\in F} f_i y_i \tag{16a}$$

$$\text{subject to} \quad \sum_{i|(i,j)\in E} x_{ij} = 1, \qquad j \in C, \tag{16b}$$

$$y_i \ge x_{ij}, \qquad (i,j) \in E, \tag{16c}$$

$$x_{ij} \ge 0, \qquad (i,j) \in E, \tag{16d}$$

$$y_i \ge 0, \qquad i \in F. \tag{16e}$$

Problem (11) can be reduced to linear program (16) by setting $F = I$, $C = S$, $E = \{(i,s) \in I \times S \mid i \in s\}$, $c_{is} = 0$ for all $(i,s) \in E$, and $f_i = |\{s \in S \mid i \in s\}|$.

THEOREM 4.3. *The constructed linear program* (16) *has optimal value at least* $|S|$, *which is attained iff variables* y_i $(i \in I)$ *satisfy* (11).

Proof. The objective value (16a) of the constructed linear program reads

$$\sum_{i\in I} f_i y_i = \sum_{i\in I} \sum_{s\in S|i\in s} y_i \ge \sum_{i\in I} \sum_{s\in S|i\in s} x_{is} = \sum_{s\in S} \sum_{i\in s} x_{is} = |S|. \tag{17}$$

The inequality in (17) is tight iff $i \in s$ implies $y_i = x_{is}$. In this case, considering (16b), for each $s \in S$ we have $1 = \sum_{i\in s} x_{is} = \sum_{i\in s} y_i$. That is, variables y_i satisfy (11).

For the other direction, suppose variables y_i satisfy (11). Set $x_{is} = y_i$ whenever $i \in s$. Now variables x_{is}, y_i form a feasible optimal solution to (16) with value $|S|$. ∎

4.5. Maximum satisfiability. Let

$$\bigwedge_{j\in C} \Bigg(\bigvee_{i\in V_j^+} v_i \vee \bigvee_{i\in V_j^-} \neg v_i \Bigg) \tag{18}$$

be a Boolean formula in conjunctive normal form with variables V and clauses C, where $V_j^+, V_j^- \subseteq V$ is the set of variables occurring nonnegated or negated, respectively, in clause $j \in C$. Let $c\colon C \to \mathbb{Q}_+$ be clause weights. The *maximum satisfiability problem* seeks to find the values of the variables to maximize the total weight of

satisfied clauses. The classical LP relaxation of this problem [36, Chapter 16] reads

$$\text{(19a)} \qquad \max \quad \sum_{j\in C} c_j z_j$$

$$\text{(19b)} \qquad \text{subject to} \quad \sum_{i\in V_j^+} x_i + \sum_{i\in V_j^-} (1-x_i) \geq z_j, \qquad j \in C,$$

$$\text{(19c)} \qquad 0 \leq z_j \leq 1, \qquad j \in C,$$

$$\text{(19d)} \qquad 0 \leq x_i \leq 1, \qquad i \in V.$$

By eliminating variables z_j, problem (19) can be written as the maximization of the concave function

$$\text{(20)} \qquad \sum_{j\in C} c_j \min\left\{1, \sum_{i\in V_j^+} x_i + \sum_{i\in V_j^-} (1-x_i)\right\}$$

over $x\colon V \to [0,1]$.

To reduce problem (11) to problem (19), let $V = I$ and let formula (18) be

$$\text{(21)} \qquad \bigwedge_{s\in S} \left(\left(\bigvee_{i\in s} v_i \right) \wedge \bigwedge_{i\in s} \neg v_i \right),$$

where each clause of type $\bigvee_{i\in s} v_i$ has weight 2 and each clause of type $\neg v_i$ has weight 1. Note that clauses of the second type can occur repeatedly—which does not matter because our formulation (18) + (19) allows repeated clauses.

THEOREM 4.4. *The constructed linear program* (19) *has optimal value at most* $\sum_{s\in S}(1+|s|)$*, which is attained iff variables* x_i $(i \in I)$ *satisfy* (11).

Proof. For the constructed problem, function (20) reads

$$\text{(22)} \qquad \sum_{s\in S} \left(2\min\left\{1, \sum_{i\in s} x_i\right\} + \sum_{i\in s}(1-x_i) \right) = \sum_{s\in S} \left(h\left(\sum_{i\in s} x_i \right) + |s| \right)$$

where the function $h\colon \mathbb{R} \to \mathbb{R}$ is given by $h(t) = 2\min\{1,t\} - t$. This function attains its maximum at $t = 1$ with value $h(1) = 1$. Therefore, function (22) attains a maximum iff $\sum_{i\in s} x_i = 1$ for all $s \in S$, with value $\sum_{s\in S}(1+|s|)$. ∎

4.6. Maximum independent set. Given a graph (V,E) with $E \subseteq \binom{V}{2}$ and vertex weights $c\colon V \to \mathbb{Q}_+$, the *maximum-weight independent set problem* seeks to find a subset $U \subseteq V$ of vertices that is independent (that is, no edge has both ends in U) and that maximizes its total weight. We consider the LP relaxation of this problem with the clique inequalities[7] [31, section 64]

$$\text{(23a)} \qquad \max \quad \sum_{i\in V} c_i x_i$$

$$\text{(23b)} \qquad \text{subject to} \quad \sum_{i\in C} x_i \leq 1, \qquad C \subseteq V \text{ is a clique},$$

$$\text{(23c)} \qquad x_i \geq 0, \qquad i \in V.$$

[7] We do not consider the (more common) LP relaxation of this problem with only cliques of size 2 (edges) because then all vertices are half-integral and hence it is not universal.

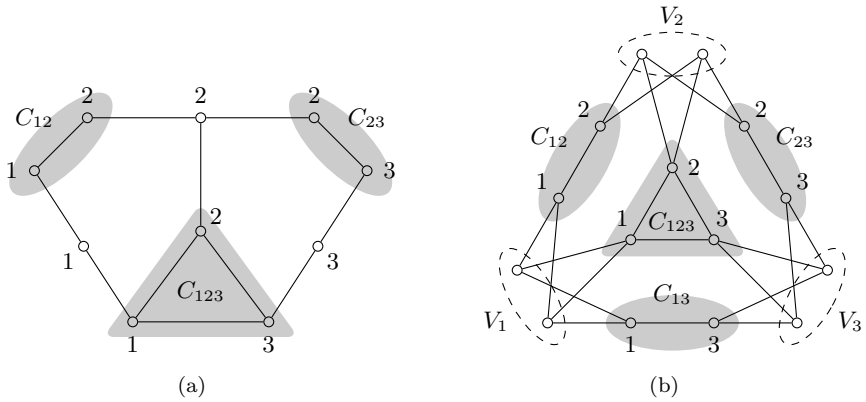

FIG. 1. *Graph (V, E) of the constructed maximum independent set problem:* (a) $I = \{1, 2, 3\}$, $S = \{\{1, 2, 3\}, \{1, 2\}, \{2, 3\}\}$, *weighted version;* (b) $I = \{1, 2, 3\}$, $S = \{\{1, 2, 3\}, \{1, 2\}, \{1, 3\}, \{2, 3\}\}$, *unweighted version.*

THEOREM 4.5. *Let $\mathcal{C} \subseteq 2^V$ be a set of (not necessarily all) cliques of the graph (V, E). Let $c_i = |\{C \in \mathcal{C} \mid i \in C\}|$. Then the optimal value of linear program* (23) *is at most $|\mathcal{C}|$, which is attained iff $\sum_{i \in C} x_i = 1$ for each $C \in \mathcal{C}$.*

Proof. For every feasible solution to (23) we have

$$\sum_{i \in V} c_i x_i = \sum_{i \in V} \sum_{C \in \mathcal{C} | i \in C} x_i = \sum_{C \in \mathcal{C}} \sum_{i \in C} x_i \leq \sum_{C \in \mathcal{C}} 1 = |\mathcal{C}|.$$

This inequality is tight iff $\sum_{i \in C} x_i = 1$ for each $C \in \mathcal{C}$. ∎

Problem (11) can be reduced to problem (23) by setting

$$\begin{aligned} V &= I \cup \{(i, s) \mid i \in s \in S\}, \\ E &= \{\{(i, s), (j, s)\} \mid i, j \in s \in S\} \cup \{\{i, (i, s)\} \mid i \in s \in S\}, \\ c_i &= |\{s \in S \mid i \in s\}|, \qquad i \in I, \\ c_{is} &= 2, \qquad i \in s \in S, \end{aligned}$$

where c_{is} is a shortcut for $c_{(i,s)}$. An example is shown in Figure 1(a). Note that the reduction is valid for any S; our example in particular has $|s| \in \{2, 3\}$.

THEOREM 4.6. *Let s_i $(i \in I)$ be such that $i \in s_i \in S$. The constructed linear program* (23) *has optimal value at most $\sum_{s \in S}(1 + |s|)$, which is attained iff variables x_{is_i} $(i \in I)$ satisfy* (11).

Proof. The weights c_i, c_{is} satisfy the assumption of Theorem 4.5 for

$$\mathcal{C} = \{\{(i, s) \mid i \in s\} \mid s \in S\} \cup \{\{i, (i, s)\} \mid i \in s \in S\}.$$

Thus the constructed linear program has optimal value at most $|\mathcal{C}| = \sum_{s \in S}(1 + |s|)$, which is attained iff its variables x_i, x_{is} satisfy

$$\sum_{i \in s} x_{is} = 1, \qquad s \in S, \tag{24a}$$

$$x_i + x_{is} = 1, \qquad i \in s \in S. \tag{24b}$$

But (24b) implies that, for each $i \in I$, variables x_{is} ($s \ni i$) are the same. Therefore, by (24a), variables x_{is_i} satisfy (11a).

For the other direction, suppose that variables x_{is_i} satisfy (11). Therefore, there exist variables x_i, x_{is} that satisfy (24); hence, they satisfy (23b) for all $C \in \mathcal{C}$. But if inequality (23b) holds for some C, then it holds also for every subset of C. As each clique that is not in $\mathcal{C}$ is a subset of some clique in $\mathcal{C}$, variables x_i, x_{is} satisfy (23b) for all cliques, and hence are feasible solutions to (23). ∎

Since $|s| \leq 3$ for all $s \in S$, the number of all cliques in the graph is $O(|S|)$. Therefore, the reduction time is linear.

4.7. Maximum independent set, unweighted. Here we present the reduction for the (unweighted) *maximum independent set problem*, that is, we have unit costs $c = 1$ in (23). As with the set cover, the reduction in this case is more complex than for the weighted version.

LEMMA 4.7. *Problem* (11) *reduces in linear time to problem* (11) *in which for each variable $i \in I$ the number n_i of its occurrences (defined in section* 3*) is divisible by* 3.

Proof. For each variable x_i ($i \in I$) in the input system (11), create new variables x'_i, x''_i. For each equation $\sum_{i \in s} x_i = 1$ ($s \in S$), create new equations $\sum_{i \in s} x'_i = 1$ and $\sum_{i \in s} x''_i = 1$. Now we have three copies of the input system (11). Therefore, for each $i \in I$ the number of occurrences of variables x_i, x'_i, x''_i is now the same. If for some $i \in I$ the number of occurrences is not divisible by 3, we increase it by 1 or 2 using the first line or both lines, respectively, of the system

$$\begin{aligned} x_i + y_i &= 1, \qquad & x'_i + y_i &= 1, \qquad & x''_i + y_i &= 1, \\ x_i + z_i &= 1, \qquad & x'_i + z_i &= 1, \qquad & x''_i + z_i &= 1, \end{aligned}$$

where y_i, z_i are auxiliary variables. ∎

LEMMA 4.8. *For every $k \in \mathbb{N}$ there exists a connected bipartite graph $(U \cup V, E)$ with partitions U and V such that $|U| = 3k$, $|V| = 2k$, $\deg(u) = 2$ for all $u \in U$, and $\deg(v) = 3$ for all $v \in V$.*

Proof. Partition the set $U \cup V$ into k subsets of size 5, each containing three vertices from U and two vertices from V. Connect the vertices in each group by edges, as follows (the bottom three nodes are from U, the top two nodes from V):

Chain the groups one by one into a cycle, using the trailing edges. ∎

Consider problem (11) in which we assume, by Lemma 4.7, that the number n_i of occurrences of each variable is divisible by 3. We now reduce this problem to the linear program (23) with unit weights. For each $i \in I$, let $(U_i \cup V_i, E_i)$ be a bipartite graph with the partitions

$$U_i = \{(i,s) \mid i \in s \in S\}, \qquad V_i = \{(i,j) \mid j \in \{1, \ldots, \tfrac{2}{3}|U_i|\}\}$$

such that $\deg(u) = 2$ for $u \in U_i$ and $\deg(v) = 3$ for $v \in V_i$. Such graphs exist by Lemma 4.8. Now set

$$V = \bigcup_{i \in I} (U_i \cup V_i), \qquad E = \{\{(i,s),(j,s)\} \mid i,j \in s \in S\} \cup \bigcup_{i \in I} E_i.$$

An example is shown in Figure 1(b).

THEOREM 4.9. *Let s_i $(i \in I)$ be such that $i \in s_i \in S$. The constructed linear program* (23) *has an optimal value of at most $\frac{1}{3}(|S| + 2\sum_{i\in I}|U_i|)$, which is attained iff variables x_{is_i} $(i \in I)$ satisfy* (11).

Proof. For each $s \in S$, the set $C_s = \{(i, s) \mid i \in s\}$ is a clique. Therefore,

$$\sum_{i\in I}\sum_{v\in U_i} x_v = \sum_{s\in S}\sum_{v\in C_s} x_v \le |S|,$$

where the inequality is tight iff $\sum_{v\in C_s} x_v = \sum_{i\in s} x_{is} = 1$.

Each vertex in U_i has two incident edges in E_i, and each vertex in V_i three incident edges in E_i. Each edge $\{u, v\} \in E_i$ is a clique. Therefore, for each $i \in I$ we have

$$2\sum_{u\in U_i} x_u + 3\sum_{v\in V_i} x_v = \sum_{\{u,v\}\in E_i} (x_u + x_v) \le 2|U_i|,$$

where the inequality is tight iff $x_u + x_v = 1$ for each $\{u, v\} \in E_i$, that is, $x_{is} + x_{ij} = 1$ for each $s \ni i$ and $j = 1, \dots, \frac{2}{3}|U_i|$.

Putting the above together, the objective function of linear program (23) reads

$$\sum_{v\in V} x_v = \frac{1}{3}\Bigg(\sum_{s\in S}\sum_{v\in C_s} x_v + \sum_{i\in I}\sum_{\{u,v\}\in E_i} (x_u + x_v)\Bigg) \le \frac{1}{3}\Bigg(|S| + 2\sum_{i\in I}|U_i|\Bigg),$$

where the inequality is tight iff

$$\sum_{i\in s} x_{is} = 1, \qquad s \in S, \tag{25a}$$

$$x_{is} + x_{ij} = 1, \qquad i \in s \in S,\ j = 1, \dots, \tfrac{2}{3}|U_i|. \tag{25b}$$

Since each graph $(U_i \cup V_i, E_i)$ is connected, (25b) implies that, for each $i \in I$, variables x_{is} $(s \ni i)$ have the same value. Hence, by (25a), variables x_{is_i} $(i \in I)$ satisfy (11).

For the other direction, suppose that variables x_{is_i} $(i \in I)$ satisfy (11). Set $x_{is} = x_{is_i}$ and $x_{ij} = 1 - x_{is}$ for all $i \in s \in S$ and $j = 1, \dots, \frac{2}{3}|U_i|$, which fulfills (25). Note that if inequality (23b) holds for some C, then it holds also for every subset of C. Since every clique is either an element of E_i or a subset of C_s, (25) implies that variables x_{is}, x_{ij} form a feasible solution to (23). ∎

4.8. Multiway cut. Let (V, E) with $E \subseteq \binom{V}{2}$ be an undirected graph with edge costs $c\colon E \to \mathbb{Q}_+$, and $T \subseteq V$ be a set of terminals. The *minimum multiway cut problem* seeks to find a subset $F \subseteq E$ of edges with minimum total cost such that in the graph $(V, E \setminus F)$ each terminal is in a different component. We consider the relaxation of this problem proposed in [6] (see also [36, Chapter 19]):

$$\min \quad \sum_{\{u,v\}\in E} \frac{c_{uv}}{2} \sum_{i\in T} |x_{ui} - x_{vi}| \tag{26a}$$

$$\text{subject to} \quad \sum_{i\in T} x_{ui} = 1, \qquad u \in V, \tag{26b}$$

$$x_{ii} = 1, \qquad i \in T, \tag{26c}$$

$$x_{ui} \ge 0, \qquad u \in V,\ i \in T. \tag{26d}$$

This is not a linear program but it can easily be transformed into one.

Although this textbook definition assumes that $c_{uv} \geq 0$ for all $\{u, v\} \in E$, it turns out that c_{uv} can be negative whenever $u \in T$ or $v \in T$. Indeed, this does not destroy the convexity of (26), by Lemma 4.10. Nonnegative costs can be recovered by replacing c_{ut} with $c_{ut} + d_u$ (where d_u are suitable constants), which does not change the set of optimal solutions. Therefore, later on we assume that $c\colon E \to \mathbb{Q}$, where $c_{uv} > 0$ whenever $u, v \notin T$.

LEMMA 4.10. *Let $v \in T$. Let x_{ui}, x_{vi} $(i \in T)$ be feasible for* (26). *Then*

$$\frac{1}{2} \sum_{i \in T} |x_{ui} - x_{vi}| = \sum_{i \in T \setminus \{v\}} x_{ui}. \tag{27}$$

Proof. By (26b)–(26d), $x_{vi} = 1$ if $i = v$ and $x_{vi} = 0$ if $i \neq v$. By (26b) and (26d), $|x_{uv} - 1| = |1 - x_{uv}| = 1 - x_{uv} = \sum_{i \in T \setminus \{v\}} x_{ui}$. Now, (27) easily follows. ∎

We will use three gadgets, described by the following theorems (see also Figure 2). Each gadget is a multiway cut instance with the optimum value of (26) equal to 0.

THEOREM 4.11. *Let gadget* $\mathsf{Unit}(u; i)$, *where $u \notin \{1, 2, 3\} \ni i$, be defined by $T = \{1, 2, 3\}$, $V = \{u\} \cup T$, $E = \{\{u, i\}\}$, and $c_{ui} = 1$. For this gadget, problem* (26) *has optimal value* 0, *attained iff $x_{ui} = 1$ and $x_{uj} = x_{uk} = 0$, where $\{i, j, k\} = \{1, 2, 3\}$.*

Proof. By Lemma 4.10, the objective (26a) for this gadget reads $x_{uj} + x_{uk}$. This attains a minimum iff $x_{ui} = 1$ and $x_{uj} = x_{uk} = 0$. ∎

THEOREM 4.12. *Let gadget* $\mathsf{Add}(u, v; i, j, k)$, *$u, v \notin \{i, j, k\} = \{1, 2, 3\}$, be defined by $T = \{1, 2, 3\}$, $V = \{u, v\} \cup T$, $E = \{\{u, v\}, \{u, i\}, \{u, j\}, \{u, k\}, \{v, i\}, \{v, j\}\}$, $c_{uv} = 2$, $c_{ui} = -2$, $c_{uj} = -3$, $c_{uk} = -1$, $c_{vi} = 3$, $c_{vj} = 4$. For this gadget, problem* (26) *has optimal value* 0, *attained iff $x_{vj} = x_{uj} + x_{uk}$ and $x_{vi} = x_{ui}$.*

Proof. Using Lemma 4.10, the objective (26a) for this gadget reads

$$\begin{aligned}
&-2(x_{uj} + x_{uk}) - 3(x_{ui} + x_{uk}) - (x_{ui} + x_{uj}) \\
&+3(x_{vj} + x_{vk}) + 4(x_{vi} + x_{vk}) + \textstyle\sum_{t=1}^{3} |x_{ut} - x_{vt}| \\
&\quad = -3x_{uj} - 5x_{uk} - 4x_{ui} + 3x_{vj} + 7x_{vk} + 4x_{vi} + \textstyle\sum_{t=1}^{3} |x_{ut} - x_{vt}| \\
&\quad = (x_{uj} - x_{vj} + |x_{uj} - x_{vj}|) + (x_{vk} - x_{uk} + |x_{uk} - x_{vk}|) + 2x_{vk} + |x_{ui} - x_{vi}|.
\end{aligned}$$

This attains minimum iff $x_{uj} \leq x_{vj}$, $x_{vk} = 0$, and $x_{ui} = x_{vi}$. In the one direction, $x_{vk} = 0$ and $x_{ui} = x_{vi}$ imply $x_{vj} = 1 - x_{vi} - x_{vk} = 1 - x_{ui} = x_{uj} + x_{uk}$. In the other direction, $x_{vj} = x_{uj} + x_{uk}$ and $x_{vi} = x_{ui}$ imply $x_{vk} = 0$ and $x_{uj} \leq x_{vj}$, so the minimum is attained iff $x_{vj} = x_{uj} + x_{uk}$ and $x_{vi} = x_{ui}$. ∎

THEOREM 4.13. *Let gadget* $\mathsf{Perm}(u, v; i, j, k)$, *where $u, v \notin \{i, j, k\} = \{1, 2, 3\}$, be defined by $T = \{1, 2, 3\}$, $V = \{u, v\} \cup T$, $E = \{\{u, v\}, \{u, j\}, \{u, k\}, \{v, j\}, \{v, k\}\}$, $c_{uv} = 2$, $c_{uj} = -4$, $c_{uk} = -1$, $c_{vj} = 4$, $c_{vk} = 3$. For this gadget, problem* (26) *has optimal value* 0, *attained iff $x_{uj} = x_{vi} = 0$, $x_{ui} = x_{vj}$, and $x_{uk} = x_{vk}$.*

Proof. Using Lemma 4.10, the objective (26a) for this gadget reads

$$\begin{aligned}
&-(x_{ui} + x_{uj}) - 4(x_{ui} + x_{uk}) + 3(x_{vi} + x_{vj}) + 4(x_{vi} + x_{vk}) + \textstyle\sum_{t=1}^{3} |x_{ut} - x_{vt}| \\
&\quad = (x_{vi} - x_{ui} + |x_{ui} - x_{vi}|) + (x_{uj} - x_{vj} + |x_{uj} - x_{vj}|) + 2x_{uj} + 2x_{vi} + |x_{uk} - x_{vk}|.
\end{aligned}$$

This attains minimum iff $x_{uj} = x_{vi} = 0$ and $x_{uk} = x_{vk}$, which implies $x_{ui} = x_{vj}$. ∎

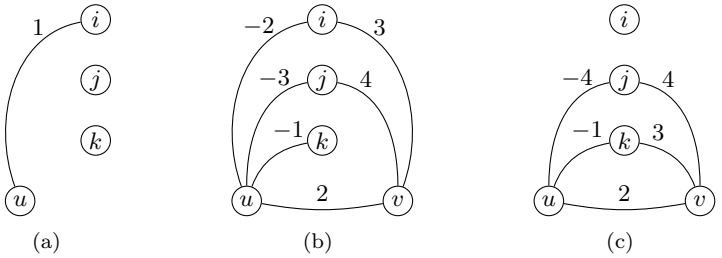

FIG. 2. *Multiway cut gadgets:* (a) $\mathsf{Unit}(u;i)$ *enforces* $x_{ui}=1$, (b) $\mathsf{Add}(u,v;i,j,k)$ *enforces* $x_{vj}=x_{uj}+x_{uk}$ *and* $x_{vi}=x_{ui}$, (c) $\mathsf{Perm}(u,v;i,j,k)$ *enforces* $x_{uj}=x_{vi}=0$, $x_{ui}=x_{vj}$, $x_{uk}=x_{vk}$.

Next, we define what it means to *combine* several multiway cut instances together. A multiway cut instance is defined by a tuple (V,E,T,c). Combining two instances (V_1,E_1,T,c_1) and (V_2,E_2,T,c_2) results in instance $(V_1\cup V_2,E_1\cup E_2,T,c_1+c_2)$, where the cost function $c_1+c_2\colon E_1\cup E_2\to\mathbb{Q}$ is given by

$$(c_1+c_2)_{uv}=\begin{cases}c_{1,uv} & \text{if } \{u,v\}\in E_1\setminus E_2,\\ c_{2,uv} & \text{if } \{u,v\}\in E_2\setminus E_1,\\ c_{1,uv}+c_{2,uv} & \text{if } \{u,v\}\in E_1\cap E_2.\end{cases}$$

LEMMA 4.14. *Let the optimal value of problem* (26) *for instances* (V_1,E_1,T,c_1), (V_2,E_2,T,c_2), *and* $(V_1\cup V_2,E_1\cup E_2,T,c_1+c_2)$ *be* y_1^*, y_2^*, *and* y^*, *respectively. Then* $y^*\geq y_1^*+y_2^*$, *which holds with the equality iff some minimizers of* (26) *for instances* (V_1,E_1,T,c_1) *and* (V_2,E_2,T,c_2) *share the values of common variables.*

Proof. The proof follows from the well-known fact that for any $f,g\colon X\to\mathbb{R}$ we have

$$\min_{x\in X}[f(x)+g(x)]\geq\min_{x\in X}f(x)+\min_{x\in X}g(x),$$

which holds with the equality iff functions f and g share a minimizer on X. ∎

We now construct a reduction from the LFE-BIN3 problem to problem (26). Initially, set $T=\{1,2,3\}$, $V=S_3\cup T$, and $E=\emptyset$. Variables x_{si} ($s\in S_3$, $i\in\{1,2,3\}$) are intended to represent the variables of (11), such that variable x_{si} of (26) equals variable $x_{\phi(s,i)}$ of (11). Equalities (11a) with $|s|=3$ are automatically enforced by (26b). In addition, we need to enforce equalities (11a) for $|s|=1$, and equalities of different variables of (26) that represent the same variable of (11). That is, we need to enforce that

$$\phi(s,i)\in I_1 \implies x_{si}=1, \tag{28a}$$

$$\phi(s,i)=\phi(t,j) \implies x_{si}=x_{tj}. \tag{28b}$$

This is done by combining the initial problem with suitable gadgets.

- Enforcing equality $x_{si}=1$ is easy, using $\mathsf{Unit}(s;i)$.
- Enforcing equality $x_{si}=x_{tj}$, with $i\neq j$, is achieved by combining gadgets $\mathsf{Add}(s,u;i,k,j)$, $\mathsf{Perm}(u,v;i,j,k)$, and $\mathsf{Add}(t,v;j,k,i)$, where u,v are new vertices and $\{i,j,k\}=\{1,2,3\}$. To attain optimum, the first gadget enforces $x_{uk}=x_{sk}+x_{sj}$ and $x_{ui}=x_{si}$, the second gadget enforces $x_{vj}=x_{ui}$ and $x_{vk}=x_{uk}$, and the third gadget enforces $x_{ti}+x_{tk}=x_{vk}$ and $x_{tj}=x_{vj}$. These equalities hold iff $x_{si}=x_{tj}$ and $x_{sj}+x_{sk}=x_{ti}+x_{tk}$.

- Enforcing $x_{si} = x_{ti}$ is achieved by combining gadgets $\mathsf{Add}(s, u; i, j, k)$ and $\mathsf{Add}(t, u; i, j, k)$, where u is a new vertex and $\{i, j, k\} = \{1, 2, 3\}$. We can verify that $x_{uj} = x_{sj} + x_{sk}$, $x_{ui} = x_{si}$ (enforced by the first gadget) and $x_{uj} = x_{tj} + x_{tk}$, $x_{ui} = x_{ti}$ (enforced by the second gadget) yield $x_{si} = x_{ti}$ and $x_{sj} + x_{sk} = x_{tj} + x_{tk}$.

For each group of n_i variables of (26) that represent the same variable x_i of (11), it suffices to enforce $n_i - 1$ equalities (28b). By Lemma 4.14, the optimal value for the constructed multiway cut instance is at least 0. To summarize, we have the following.

THEOREM 4.15. *Let s_i, j_i $(i \in I)$ be such that $\phi(s_i, j_i) = i$. The constructed problem* (26) *has an optimal value at least* 0*, which is attained iff variables $x_{s_i j_i}$ $(i \in I)$ satisfy* (11)*.*

5. Strong reductions to LP relaxations. Here we present reductions from the LFE-BIN3 problem to the LP relaxations of three more combinatorial problems, thereby proving Theorem 2.7. In contrast to section 4, here the solution set of an LFE-BIN3 instance is a coordinate projection of the feasible (rather than optimal) set of each LP relaxation.

5.1. Exact set cover. The *exact set cover problem* is a classical NP-complete decision problem [19], which seeks to find a set cover (see section 4.1) that covers each element exactly once (that is, a partition of the ground set). Its LP relaxation has the form $Ax = 1$, $x \geq 0$ (with no objective function), where A is as in section 4.1. By endowing it with a linear objective function, we obtain the LP relaxation of the *minimum-cost exact set cover problem*, which seeks to find an exact cover with minimum total cost.

5.2. Three-dimensional matching/assignment. Given a finite set V and a set of triplets $E \subseteq V^3$, a subset $F \subseteq E$ is a *three-dimensional (3-D) matching* if no two elements of F agree in any coordinate. The *3-D (exact) matching problem* (see, e.g., [19]) seeks to find a 3-D matching such that $|F| = |V|$. Its LP relaxation reads [3]

$$\sum_{j,k|(i,j,k)\in E} x_{ijk} = 1, \qquad i \in V, \tag{29a}$$

$$\sum_{i,k|(i,j,k)\in E} x_{ijk} = 1, \qquad j \in V, \tag{29b}$$

$$\sum_{i,j|(i,j,k)\in E} x_{ijk} = 1, \qquad k \in V, \tag{29c}$$

$$x_{ijk} \geq 0, \qquad (i, j, k) \in E. \tag{29d}$$

An equality (11a) with $|s| = 3$ can be encoded with a single gadget (see Figure 3(a)), which is an instance of system (29) defined by

$$V = \{0, 1, 2, 3\}, \qquad E = \bigcup_{i\in\{1,2,3\}} \{(0, i, i), (i, 0, 0), (i, i, i)\}. \tag{30}$$

For this instance, equalities (29a)–(29c) read

$$x_{011} + x_{022} + x_{033} = x_{100} + x_{200} + x_{300} = 1,$$
$$x_{0ii} + x_{iii} = x_{i00} + x_{iii} = 1, \qquad i \in \{1, 2, 3\}.$$

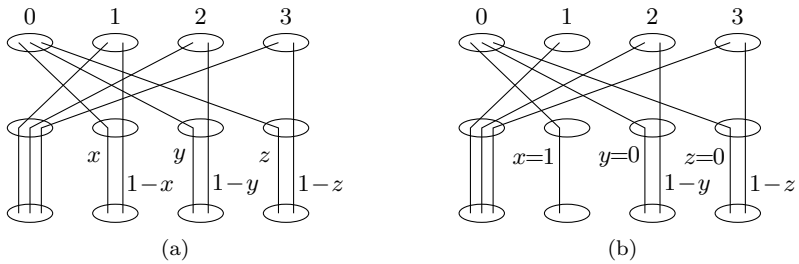

FIG. 3. *The three-dimensional matching gadgets.* (a) *Any solution to* (29) *fulfills* $x+y+z=1$, *where* x, y, z *are shortcuts for* x_{011}, x_{022}, x_{033}, *respectively. Each polyline corresponds to one element of* E *and each ellipse corresponds to one equation* (29a)–(29c). (b) *If edge* $(1,1,1)$ *is removed, then any solution to* (29) *fulfills* $x=1$ *and* $y=z=0$.

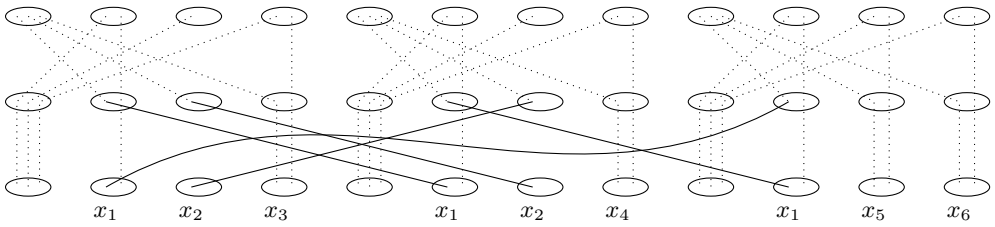

FIG. 4. *The three-dimensional matching instance representing the system* $x_1+x_2+x_3=x_1+x_2+x_4=x_1+x_5+x_6=1$, $x_2=1$, $x_1,\dots,x_6\geq 0$. *The dotted edges are the original gadget edges. The solid edges enforce the equality of three occurrences of* x_1 *and of two occurrences of* x_2 *(they correspond to the cyclic permutations* π_k*).*

This gadget can be easily modified to also encode an equality (11a) with $|s|=1$: by omitting edge (i,i,i) for some $i\in\{1,2,3\}$ (see Figure 3(b)), it enforces $x_{0ii}=1$.

To construct a reduction from the LFE-BIN3 problem to problem (29), we create an instance of system (29) consisting of one copy of gadget (30) for each element of S_3. To ensure that the variables of (29) representing the same variable of (11) attain the same value, we rewire some of the links in the bottom part of the gadgets (see Figure 4 for an example). To this end, for each $i\in I$ we choose a cyclic permutation π_i on the set $\phi^{-1}(i)$, i.e., such that the orbit of any element of $\phi^{-1}(i)$ under π_i is the whole $\phi^{-1}(i)$. We write $\pi(s,i)$ as a shortcut for $\pi_{\phi(s,i)}(s,i)$. The resulting instance is defined as

$$V=S_3\times\{0,1,2,3\},\qquad E=\bigcup_{s\in S_3}\bigcup_{i\in\{1,2,3\}}E_{si},$$

$$E_{si}=\{((s,0),(s,i),\pi(s,i)),((s,i),(s,0),(s,0))\}\cup\{((s,i),(s,i),(s,i))\mid\phi(s,i)\notin I_1\}.$$

To prove correctness, we show that, for every $s\in S_3$ and $i\in\{1,2,3\}$, variable $x_{s0,si,\pi(s,i)}$ of (29) represents variable $x_{\phi(s,i)}$ of (11), where $x_{si,sj,sk}$ is a shortcut for $x_{(s,i),(s,j),(s,k)}$. That is,

$$\phi(s,i)\in I_1 \implies x_{s0,si,\pi(s,i)}=1, \tag{31a}$$

$$\phi(s,i)=\phi(t,j) \implies x_{s0,si,\pi(s,i)}=x_{t0,tj,\pi(t,j)}. \tag{31b}$$

Consider $s,t\in S_3$ and $i,j\in\{1,2,3\}$ such that $\pi(s,i)=(t,j)$.

- If $((t,j),(t,j),(t,j)) \in E$ (i.e., $\phi(t,j) \notin I_1$), then (29c) + (29b) reads

$$x_{s0,si,\pi(s,i)} + x_{tj,tj,tj} = x_{t0,tj,\pi(t,j)} + x_{tj,tj,tj} = 1.$$

  This implies $x_{s0,si,\pi(s,i)} = x_{t0,tj,\pi(t,j)}$.
- If $((t,j),(t,j),(t,j)) \notin E$ (i.e., $\phi(t,j) \in I_1$), then the equations have the form $x_{s0,si,\pi(s,i)} = x_{t0,tj,\pi(t,j)} = 1$.

Since $\bigcup S_3 = I$ (see section 3) and $\pi_{\phi(s,i)}$ cycles through all representatives of variable $x_{\phi(s,i)}$ in (11), implications (31) hold. We have the following.

THEOREM 5.1. *Let s_i, j_i $(i \in I)$ be such that $\phi(s_i, j_i) = i$. Then variables $x_{s_i 0, s_i j_i, \pi(s_i,j_i)}$ $(i \in I)$ satisfy* (11) *iff they can be extended to a solution to* (29).

In other words, the solution set of system (11) is a coordinate projection of the solution set of the constructed system (29).

By endowing system (29) with a linear objective function, we obtain the LP relaxation of a weighted version of the 3-D matching problem, known as the 3-*D assignment problem* [3].

5.3. Constraint satisfaction. Let (V,E) be an undirected graph, where V is a set of variables, and $E \subseteq \binom{V}{2}$ a set of variable pairs. Let D be a finite *domain* of the variables. The (binary) *constraint satisfaction problem* (CSP) [23, 10] seeks an assignment $\lambda\colon V \to D$ to the variables such that

$$\lambda(u) \in D_u, \qquad u \in V, \tag{32a}$$

$$(\lambda(u), \lambda(v)) \in D_{uv}, \qquad \{u,v\} \in E, \tag{32b}$$

where $D_u \subseteq D$ and $D_{uv} \subseteq D \times D$ are given unary and binary relations, and we adopt the convention that $(i,j) \in D_{uv}$ iff $(j,i) \in D_{vu}$. This problem has the natural LP relaxation[8]

$$\sum_{j \in D} x_{uv}(i,j) = x_u(i), \quad u \in V,\ v \in E_u,\ i \in D, \tag{33a}$$

$$\sum_{i \in D} x_u(i) = 1, \qquad u \in V, \tag{33b}$$

$$x_u(i) \geq 0, \qquad u \in V,\ i \in D, \tag{33c}$$

$$x_{uv}(i,j) \geq 0, \qquad \{u,v\} \in E,\ (i,j) \in D \times D, \tag{33d}$$

$$x_u(i) = 0, \qquad u \in V,\ i \in D \smallsetminus D_u, \tag{33e}$$

$$x_{uv}(i,j) = 0, \qquad \{u,v\} \in E,\ (i,j) \in (D \times D) \smallsetminus D_{uv}, \tag{33f}$$

where $E_u = \{v \in V \mid \{u,v\} \in E\}$ denote the neighbors of variable $u \in V$, and we again adopt the convention that $x_{uv}(i,j) = x_{vu}(j,i)$.

We will use two gadgets (see Figure 5). They are instances of problem (33) defined by $V = \{u,v\}$, $D = \{1,2,3\}$, $E = \{\{u,v\}\}$, $D_u = D_v = D$, and D_{uv} given as follows.

- Gadget $\mathsf{Copy}(u,v;i,j)$, where $i,j \in \{1,2,3\}$, has

$$D_{uv} = \{(i,j)\} \cup \{(k,l) \mid k \in \{1,2,3\} \smallsetminus \{i\},\ l \in \{1,2,3\} \smallsetminus \{j\}\}.$$

  This enforces $x_u(i) = x_v(j)$.

[8] This LP relaxation of CSP is folklore, so we do not give any references here. But it is a special case of the LP relaxation of the valued CSP, for which we give references later.

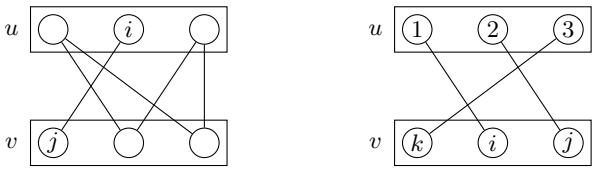

FIG. 5. *CSP gadgets:* (a) $\mathsf{Copy}(u, v; i, j)$ *enforces* $x_u(i) = x_v(j)$, (b) $\mathsf{Perm}(u, v; i, j, k)$ *enforces* $x_u(1) = x_v(i)$, $x_u(2) = x_v(j)$, $x_u(3) = x_v(k)$. *The boxes, circles, and edges depict elements of* V, D, *and* D_{uv}, *respectively (as in* [39, 28]*).*

- Gadget $\mathsf{Perm}(u, v; i, j, k)$, where $\{i, j, k\} = \{1, 2, 3\}$, has

$$D_{uv} = \{(1, i), (2, j), (3, k)\}.$$

This enforces $x_u(1) = x_v(i)$, $x_u(2) = x_v(j)$, $x_u(3) = x_v(k)$.

We omit the proofs of the claimed properties of the gadgets because they are easy.

We now reduce the LFE-BIN3 problem to problem (33). We assume that $\bigcup S_3 = I$ (see section 3). Initially, set $D = \{1, 2, 3\}$, $V = S_3$, $E = \emptyset$, and $D_s = D$ for each $s \in S_3$. Variables $x_s(i)$ are intended to represent the variables of (11), such that variable $x_s(i)$ of (33) equals variable $x_{\phi(s,i)}$ of (11). Equalities (11a) for $|s| = 3$ hold by (33b). In addition, we need to enforce equalities (11a) for $|s| = 1$ and equalities of different variables of (33) that represent the same variable of (11). That is, we need to enforce that

$$\phi(s, i) \in I_1 \implies x_s(i) = 1, \tag{34a}$$

$$\phi(s, i) = \phi(t, j) \implies x_s(i) = x_t(j). \tag{34b}$$

This is done as follows.

- Equality $x_s(i) = 1$ is enforced by changing $D_s = D$ to $D_s = \{i\}$.
- If $|s \cap t| = 1$, equality $x_s(i) = x_t(j)$ is enforced by $\mathsf{Copy}(s, t; i, j)$.
- If $|s \cap t| > 1$, equalities $x_s(i) = x_t(j)$ are enforced by Perm. Note here that if the two equations (11a) share two variables, their third variables have the same value (e.g., $x_1 + x_2 + x_3 = 1$ and $x_1 + x_2 + x_4 = 1$ implies $x_3 = x_4$).

For each group of n_i variables of (33) that represent the same variable x_i of (11), it suffices to enforce only $n_i - 1$ equalities (34b). We have the following.

THEOREM 5.2. *Let* s_i, j_i $(i \in I)$ *be such that* $\phi(s_i, j_i) = i$. *Variables* $x_{s_i}(j_i)$ $(i \in I)$ *satisfy* (11) *iff they can be extended to a solution to* (33).

The CSP has a weighted version, known under several names, such as the *valued CSP* [24, 37], *discrete energy minimization* [17], *MAP inference in graphical models* [38], and the *max-sum labeling problem* [39]. Its LP relaxation [33, 39, 20, 4] is obtained by endowing system (33) with a linear objective function.

6. P-completeness. Our (nearly) linear time reductions show that, for sequential algorithms, many LP relaxations are as hard as the general LP problem. Here we show they are hard also for parallel algorithms, since they are P-complete under log-space reductions.

The LP problem is P-complete under log-space reductions, by reduction from the Boolean circuit value problem (CVP) [30, 35]. A reduction in [16] even implies P-completeness for linear programs with coefficients $\{-1, 0, 1\}$. By modifying

these known reductions, it is straightforward to show that the LFE-BIN3 problem is P-complete, which in turn implies P-completeness of our LP relaxations.

THEOREM 6.1. *The LFE-BIN3 problem is P-complete under log-space reduction.*

Proof. The reduction from the (P-complete) CVP to the LP problem proposed in [35, (LP1)] can be rewritten, using slack variables, to an LFE with coefficients $\{-1, 0, 1\}$. This LFE reduces to LFE-BIN3, by Theorems 3.5 and 3.8. Since the variables of the LFE take values from $[0, 1]$ and each equation has at most two variables with coefficient -1 and at most two variables with coefficient 1, in Theorem 3.8 it suffices to use the scale $\sigma = \frac{1}{2}$. Such a reduction can be done deterministically in logarithmic space. ∎

THEOREM 6.2. *Each LP relaxation from sections* 4 *and* 5 *is P-complete under log-space reductions.*

Proof. We presented the reductions in sections 4 and 5 as reductions in linear time. These reductions can be modified such that they need only deterministic logarithmic space[9] (but not necessarily linear time). Indeed, it is easy to check that each reduction can be performed using a constant number of counters. Thus, by Theorem 6.1, the LP relaxations are P-complete under log-space reductions. ∎

It is known that even approximating the general LP problem (i.e., finding a feasible point with the objective value near to the optimum value) is P-complete under log-space reductions [32]. The proof considers the problem of computing the number of true gates in the Boolean circuit with bounded fan-out/fan-in gates. This result implies the following theorem.

THEOREM 6.3. *To approximate any of the LP relaxations from section* 5 *within any fixed approximation factor is P-complete under log-space reductions.*

Proof. The problem of counting true gates in the circuit can be formulated as an LP problem [32]. This LP problem reduces to LP/LFE-BIN3 with the scale σ bounded by a constant independent on the circuit size (because each variable takes values from $[0, 1]$ and each constraint has a bounded number of variables). By Theorem 2.7, LP/LFE-BIN3 reduces further to each LP relaxation from section 5. And again, all the considered reductions can be done in deterministic logarithmic space. ∎

The situation is different for the LP relaxations from section 4. The LP relaxations of set cover, set packing and maximum satisfiability belong to the class of positive linear programs (PLP),[10] for which efficient approximation parallel algorithms exist [22, 35, 34].

7. Comments. In this final section we give some additional comments on the presented results, anticipating possible questions by the reader.

7.1. Consequences for designing algorithms. Arguably, the most important consequences of our reductions are constraints on algorithms to solve the LP relaxations. Leaving runtime aside, they show that such algorithms cannot be arbitrarily simple since they must be able to solve any linear program. Considering runtime, the nearly linear time of the reductions implies that the size of the output LP relaxation is nearly linear in the size of the input linear program. Thus, any (worst case) lower

[9] Unlike for linear-time reductions, here the Turing machine suffices as the computational model.

[10] A linear program is a PLP if it has the form $\min\{c^T x \mid Ax \geq b,\ x \geq 0\}$ or $\max\{c^T x \mid Ax \leq b,\ x \geq 0\}$, where all entries of A, b, c are nonnegative.

bound on time complexity of the LP problem (measured only by its size (1)) is inherited by such algorithms. In this sense, solving the LP relaxations is not easier than solving any linear program.

Let us emphasize that this statement assumes that solving different LP instances of the same size takes comparable time. In reality, runtimes of LP algorithms to some extent depend not only on the instance size but also, e.g., on the number of variables and constraints. But the nearly linear time of our reductions does not imply that the number of variables/constraints of the output LP relaxation is linear in the number of variables/constraints of the input linear program, e.g., when an LFE problem $kx = 1$, $x \geq 0$ (which has one variable and one equation with coefficient $k \in \mathbb{N}$) is reduced to the LFE-BIN3 problem as described in section 3, the result has $\Theta(\log k)$ variables and equations.

The LP algorithms can, moreover, be affected by the scale σ, introduced in Theorem 3.8. In general, the magnitude of σ is exponentially small (but its number of bits is still linear in the input size), and hence the variable values in the solved LP problem can be negligible relative to the coefficients. Note, however, that for some input problems it is possible to find a much tighter bound on vertex coordinates than the one given by Lemma 3.7 and hence on a larger scale (an example is in the proof of Theorem 6.1).

Our results imply that any two of the LP relaxations considered reduce to each other in nearly linear time. At least for some of these reductions, the number of output variables and constraints is linear in the number of input ones, because LP relaxations have small (bounded) coefficients and so the system (8) will have constant size.

7.2. Easy vs. hard LP relaxations. What LP relaxations are easy, such that the LP problem cannot be reduced to them in nearly linear time? Allowing general reductions (in which solutions to input instances can be obtained from solutions to output instances by any nearly linear-time algorithm, see e.g., [36, section A.3.1]), this question is ill-posed as we cannot exclude that the LP problem is solvable in nearly linear time.

Our reductions are not general because solutions to output instances are mapped to solutions to input instances by simple affine maps. Restricting ourselves to such reductions, we can ask which LP relaxations are such that their optimal set cannot be an arbitrary polytope (up to an affine map). These include LP relaxations such that the vertices of their feasible set cannot be arbitrary fractions. One such class is formed by half-integral LP relaxations of problems such as vertex cover [36] or node multiway cut [11]. Another example is the knapsack problem, whose LP relaxation always has an optimum solution with at most one fractional component [7]. Another potential candidate is linear optimization over the metric polytope (an LP relaxation of the max-cut problem) [9]. Its vertices can have arbitrary (other than 2) denominators [21, Proposition 4.1]. However, to construct a vertex with denominator d, a graph with $n \geq 3d - 1$ vertices is needed in the proof. It is open if a graph with $O(\log d)$ vertices would suffice.

7.3. Weak vs. strong reductions. Similarly, one can ask why some LP relaxations allow only weak reduction while some allow a strong one. One distinction is that, for the LP relaxations from section 4, finding a feasible solution is trivial but finding an optimal solution is hard, e.g., the covering linear program (12) is feasible iff each row of A is nonzero, and in that case it has a trivial feasible solution $x = 1$. For the LP relaxations from section 5, finding a feasible solution is already hard.

Another distinction is that feasible sets of the LP relaxations from section 4 are not universal (i.e., affinely equivalent to any polytope) but their optimal sets are, while feasible sets of LP relaxations from section 5 are universal.

One might think that there is no qualitative difference between the weak and strong reductions because it is possible to optimize over a polytope face by adding a suitable multiple of the face-supporting objective to the final objective. However, to reduce the general linear program in this way requires the multiple to be very large, which might prevent any benefit from approximate solutions of the obtained linear program.

7.4. Relation to extension complexity. Our results can be related to extension complexity theory (see, e.g., [5]), which seeks to simplify descriptions of integral hulls of combinatorial problems by representing them as projections of other polytopes (i.e., by introducing auxiliary variables). A polytope Q is an extended formulation of a polytope P if P is an affine projection of Q. The size $|Q|$ of Q is defined as the number of facets of Q. The extension complexity of P, $\mathrm{xc}(P)$, is the smallest size of an extended formulation of P.

One can consider a restricted version of extended complexity, $\mathrm{xc}_F(P)$, by requiring the polytopes Q to be from some family F of polytopes. We assume that F is universal, i.e., every polytope is a projection of some polytope from F. In our case, F can be the solution sets of the LFE-BIN3 problem or the optimal sets of some LP relaxation from sections 4 or 5. If $P = \{x \mid Ax \leq b\}$, we denote $L(A, b)$ by $L(P)$ (where L was defined in (1)). We have $L(P) \geq |P|$ because $L(P)$ takes into account the sizes of coefficients while $|P|$ only counts (nonredundant) inequalities.

Clearly, $\mathrm{xc}(P) \leq |P|$. The reductions presented imply $\mathrm{xc}_F(P) = O(L(P))$, but $\mathrm{xc}_F(P) \leq |P|$ generally holds only if $P \in F$, e.g., for $P = \{x \in \mathbb{R} \mid kx = 1,\ x \geq 0\}$ (where $k \in \mathbb{N}$) we have $|P| = 1$ but $\mathrm{xc}_F(P) = \Theta(\log k)$. On the other hand, the permutohedron P_{perm} of order n fulfills $|P_{\mathrm{perm}}| = \Theta(2^n)$ and $\mathrm{xc}(P_{\mathrm{perm}}) = \Theta(n \log n)$ [12]. It has an extension Q such that $L(Q) = O(n^2 \log n)$, and hence $\mathrm{xc}_F(P_{\mathrm{perm}}) < |P_{\mathrm{perm}}|$. To summarize, we have

$$\mathrm{xc}(P) \leq |P| \leq L(P), \tag{35a}$$

$$\mathrm{xc}(P) \leq \mathrm{xc}_F(P) = O(L(P)), \tag{35b}$$

while $|P|$ and $\mathrm{xc}_F(P)$ are incomparable. In future research, it might be valuable to better understand the relations between $\mathrm{xc}(P)$, $\mathrm{xc}_F(P)$, $|P|$, and $L(P)$ for interesting families F and families of polytopes P.

Acknowledgment. We thank the anonymous reviewers for many valuable comments.

REFERENCES

[1] L. J. Billera and A. Sarangarajan, *All* 0-1 *polytopes are traveling salesman polytopes*, Combinatorica, 16 (1996), pp. 175–188.

[2] Y. Boykov and V. Kolmogorov, *An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision*, IEEE Trans. Pattern Anal. Mach. Intell., 26 (2004), pp. 1124–1137.

[3] R. E. Burkard, *Selected topics on assignment problems*, Discrete Appl. Math., 123 (2002), pp. 257–302.

[4] 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., 18 (2004), pp. 608–625.

[5] M. Conforti, G. Cornuéjols, and G. Zambelli, *Extended formulations in combinatorial optimization*, 4OR, 8 (2010), pp. 1–48.
[6] G. Călinescu, H. Karloff, and Y. Rabani, *An improved approximation algorithm for multiway cut*, in Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing, Association for Computing Machinery, New York, 1998, pp. 48–52.
[7] G. B. Dantzig, *Discrete-variable extremum problems*, Oper. Res., 5 (1957), pp. 266–288.
[8] J. A. De Loera and S. Onn, *All linear and integer programs are slim 3-way transportation programs*, SIAM J. Optim., 17 (2006), pp. 806–821.
[9] M. M. Deza and M. Laurent, *Geometry of Cuts and Metrics*, Algorithms Combin. 15, Springer, Berlin, 1997.
[10] E. Freuder and A. K. Mackworth, *Constraint satisfaction: An emerging paradigm*, in Handbook of Constraint Programming, Elsevier Science, New York, 2006, pp. 13–28.
[11] N. Garg, V. V. Vazirani, and M. Yannakakis, *Multiway cuts in node weighted graphs*, J. Algorithms, 50 (2004), pp. 49–61.
[12] M. X. Goemans, *Smallest compact formulation for the permutahedron*, Math. Program., 153 (2015), pp. 5–11.
[13] A. V. Goldberg, S. Hed, H. Kaplan, R. E. Tarjan, and R. F. Werneck, *Maximum flows by incremental breadth-first search*, in Algorithms: ESA 2011, C. Demetrescu and M. M. Halldórsson, eds., Lecture Notes in Comput. Sci. 6942, Springer, Heidelberg, 2011, pp. 457–468.
[14] Y. Gurevich and S. Shelah, *Nearly linear time*, in Logic at Botik 1989, Lecture Notes in Comput. Sci. 363, Springer, Heidelberg, 1989, pp. 108–118.
[15] D. S. Hochbaum, *Monotonizing linear programs with up to two nonzeroes per column*, Oper. Res. Lett., 32 (2004), pp. 49–58.
[16] A. Itai, *Two-commodity flow*, J. ACM, 25 (1978), pp. 596–611.
[17] J. H. Kappes, B. Andres, F. A. Hamprecht, C. Schnörr, S. Nowozin, D. Batra, S. Kim, B. X. Kausler, T. Kröger, J. Lellmann, N. Komodakis, B. Savchynskyy, and C. Rother, *A comparative study of modern inference techniques for structured discrete energy minimization problems*, Int. J. Comput. Vis., 115 (2015), pp. 155–184.
[18] N. Karmarkar, *A new polynomial-time algorithm for linear programming*, in Proceedings of the Sixteenth Annual ACM Symposium on Theory of Computing, Association for Computing Machinery, New York, 1984, pp. 302–311.
[19] R. Karp, *Reducibility among combinatorial problems*, in Complexity of Computer Computations, R. Miller and J. Thatcher, eds., Plenum Press, New York, 1972, pp. 85–103.
[20] V. Kolmogorov, J. Thapper, and S. Zivny, *The power of linear programming for general-valued CSPs*, SIAM J. Comput., 44 (2015), pp. 1–36.
[21] M. Laurent and S. Poljak, *One-third-integrality in the max-cut problem*, Math. Program., 71 (1995), pp. 29–50.
[22] M. Luby and N. Nisan, *A parallel approximation algorithm for positive linear programming*, in Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing, Association for Computing Machinery, New York, 1993, pp. 448–457.
[23] A. Mackworth, *Constraint satisfaction*, in Encyclopedia of Artificial Intelligence, S. Shapiro, ed., John Wiley, New York, 1991, pp. 285–292.
[24] P. Meseguer, F. Rossi, and T. Schiex, *Soft constraints*, in Handbook of Constraint Programming, Elsevier, New York, 2006, pp. 281–328.
[25] A. V. Naik, K. W. Regan, and D. Sivakumar, *On quasilinear-time complexity theory*, Theoret. Comput. Sci., 148 (1995), pp. 325–349.
[26] C. H. Papadimitriou and K. Steiglitz, *Combinatorial Optimization: Algorithms and Complexity*, Dover, New York, 1998.
[27] D. Průša and T. Werner, *LP relaxations of some NP-hard problems are as hard as any LP*, in Proceedings of the 28th Annual ACM-SIAM Symposium on Discrete Algorithms, Society for Industrial and Applied Mathematics, Philadelphia, PA, 2017, pp. 1372–1382.
[28] D. Průša and T. Werner, *Universality of the local marginal polytope*, IEEE Trans. Pattern Anal. Mach. Intell., 37 (2015), pp. 898–904.
[29] D. Průša and T. Werner, *LP relaxation of the Potts labeling problem is as hard as any linear program*, IEEE Trans. Pattern Anal. Mach. Intell., 39 (2017), pp. 1469–1475.
[30] G. Raymond, H. J. Hoover, and W. L. Ruzzo, *A compendium of problems complete for P*, Technical report TR 91-05-01, University of Washington, Seattle, WA, 1991.
[31] A. Schrijver, *Combinatorial Optimization: Polyhedra and Efficiency*, Algorithms Combin. 24, Springer, Heidelberg, 2003.
[32] M. Serna, *Approximating linear programming is log-space complete for P*, Inform. Process. Lett., 37 (1991), pp. 233–236.

[33] M. I. Shlezinger, *Syntactic analysis of two-dimensional visual signals in noisy conditions*, Cybernet. Systems Anal., 12 (1976), pp. 612–628 (in English); Kibernetika, 4 (1976), pp. 113—130 (in Russian).
[34] L. Trevisan, *Parallel approximation algorithms by positive linear programming*, Algorithmica, 21 (1998), pp. 72–88.
[35] L. Trevisan and F. Xhafa, *The parallel complexity of positive linear programming*, Parallel Process. Lett., 8 (1998), pp. 527–533.
[36] V. V. Vazirani, *Approximation Algorithms*, Springer, New York, 2001.
[37] S. Živný, *The Complexity of Valued Constraint Satisfaction Problems*, Cogn. Technol., Springer, Heidelberg, 2012.
[38] M. J. Wainwright and M. I. Jordan, *Graphical models, exponential families, and variational inference*, Found. Trends. Mach. Learn., 1 (2008), pp. 1–305.
[39] T. Werner, *A linear programming approach to max-sum problem: A review*, IEEE Trans. Pattern Anal. Mach. Intell., 29 (2007), pp. 1165–1179.