View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Elsevier - Publisher Connector

Discrete Applied Mathematics 160 (2012) 2419–2434

Contents lists available at SciVerse ScienceDirect

Discrete Applied Mathematics

journal homepage: www.elsevier.com/locate/dam

Generalized roof duality

Fredrik Kahl, Petter Strandmark *

Centre for Mathematical Sciences, Lund University, Sweden

ARTICLE INFO

Article history:
Received 1 October 2011
Received in revised form 14 June 2012
Accepted 15 June 2012
Available online 12 July 2012

Keywords:
Roof duality
Higher-order
MRF
Computer vision

ABSTRACT

The roof dual bound for quadratic unconstrained binary optimization is the basis for several methods for efficiently computing the solution to many hard combinatorial problems. It works by constructing the tightest possible lower-bounding submodular function, and instead of minimizing the original objective function, the relaxation is minimized. However, for higher-order problems the technique has been less successful. A standard technique is to first reduce the problem into a quadratic one by introducing auxiliary variables and then apply the quadratic roof dual bound, but this may lead to loose bounds.

We generalize the roof duality technique to higher-order optimization problems. Similarly to the quadratic case, optimal relaxations are defined to be the ones that give the maximum lower bound. We show how submodular relaxations can efficiently be constructed in order to compute the generalized roof dual bound for general cubic and quartic pseudo-boolean functions. Further, we prove that important properties such as persistency still hold, which allows us to determine optimal values for some of the variables. From a practical point of view, we experimentally demonstrate that the technique outperforms the state of the art for a wide range of applications, both in terms of lower bounds and in the number of assigned variables.

© 2012 Elsevier B.V. All rights reserved.

1. Introduction

Consider a pseudo-boolean function $f : \mathbf{B}^n \to \mathbf{R}$ where $\mathbf{B} = \{0, 1\}$ and suppose f is represented by a multilinear polynomial of the form

$$f(\boldsymbol{x}) = \sum_i a_i x_i + \sum_{i<j} a_{ij} x_i x_j + \sum_{i<j<k} a_{ijk} x_i x_j x_k + \cdots, \tag{1}$$

of $\mathrm{degree}(f) = m$. In this paper, we are interested in the following optimization problem:

$$\min_{\boldsymbol{x} \in \mathbf{B}^n} f(\boldsymbol{x}). \tag{2}$$

This problem is well-known to be NP-hard, so we have to settle for non-optimal solutions. The purpose of this paper is to provide means to compute effective bounds on the optimal value for large-scale problems (that is, when n is of the order of several thousands of variables). We will use submodular relaxations for bounding as submodular function minimization has polynomial time complexity [18,24]. We will primarily investigate the cases when $m = 2$, $m = 3$ or $m = 4$. Our framework applies for arbitrary degree problems, but there are still missing building blocks in order to devise an algorithm for $m \geq 5$.

* Corresponding author. Tel.: +46 46 2224747; fax: +46 46 2224010.
E-mail addresses: fredrik@maths.lth.se (F. Kahl), petter@maths.lth.se (P. Strandmark).
URLs: http://www.maths.lth.se/matematiklth/personal/fredrik (F. Kahl), http://www.maths.lth.se/matematiklth/personal/petter (P. Strandmark).

0166-218X/$ – see front matter © 2012 Elsevier B.V. All rights reserved.
doi:10.1016/j.dam.2012.06.009

There are numerous application problems that can be cast in this framework, ranging from portfolio problems in operations research to the minimization of the Ising model in physics. Many graph-theoretic problems can also be turned into a pseudo-boolean optimization problem, for example, maximum satisfiability and vertex cover [3,12]. In this work, we have been motivated by the many applications in computer vision and machine learning. State-of-the-art methods for stereo, segmentation and image denoising are often formulated as the inference of the maximum a posteriori estimate, which can be cast as a minimization problem where the objective function is given by a pseudo-boolean function [19].

Naturally, there have been many approaches for solving such optimization problems, especially for quadratic ($m = 2$) pseudo-boolean problems. One of the most successful bounds in terms of computational efficiency is the "roof dual" of a quadratic pseudo-boolean optimization problem, introduced in [13]. The idea is to relax the original problem and then compute a bound on the optimal value with a polynomial time algorithm. More specifically, it was shown that three different types of linear programming relaxations of quadratic pseudo-boolean problems yield the same bound—the roof dual. We show that the same bound is attained with submodular relaxations. Further, it was shown that partial solutions can be extracted from the relaxed solutions, a property known as *persistency*. In subsequent studies the technique has been refined and roof duality has shown to produce state-of-the-art results for a variety of application problems compared to other bounding techniques based on linear programming and semidefinite relaxations; see [2–4,19]. A key advantage is that max-flow/min-cut computations can be applied to an appropriately constructed graph for quadratic pseudo-boolean polynomials [3].

The primary focus of this paper is to generalize the roof duality framework for higher-order pseudo-boolean functions. Our main contributions are (i) how one can define a general bound for any order (for which the quadratic case is a special case) and (ii) how one can efficiently compute solutions that attain this bound in polynomial time. These contributions are of course coupled—it makes little sense to define a bound that is not tractable. In addition, we show that persistency is preserved so fixation of some of the variables to optimal values is possible.

1.1. Related work

In recent years, there has been an increasing interest in higher-order models and approaches for minimizing the corresponding objective functions in computer vision and machine learning. For example, in [22], approximate belief propagation is used with a learned higher-order model for image denoising. Similarly, in [7], a higher-order model is learned for texture restoration, but the model is restricted to submodular energies which can be optimized exactly in polynomial time. Curvature regularization requires higher-order models [29,30]. Even global potentials defined over all variables have been considered, for example, in [26] for ensuring connectedness, in [21] to model co-occurrence statistics of objects. Another state-of-the-art example is [33] where second-order surface priors are used for stereo reconstruction. The optimization strategies rely on dual decomposition [20,31], move-making algorithms [17,23], linear programming [32], belief propagation [22] and, of course, max-flow/min-cut.

The inspiration for our work comes primarily from three different sources. First of all, as max-flow/min-cut computations are considered to be state-of-the-art for quadratic pseudo-boolean polynomials [4,19], reduction techniques of higher-order polynomials ($m > 2$) have been explored, for example, [25,9,28,15,11,8]. However, all of these approaches choose suboptimally between a fixed set of possible reductions. Then, there exist several suggestions for generalizations of roof duality for higher-order polynomials. In [25], a roof duality framework is presented based on reduction, but at the same time, the authors note that their roof duality bound depends on which reductions are applied. In [18], submodular relaxations are proposed as a generalization for roof duality, but no method is given for constructing or minimizing such relaxations. Our framework also builds on using submodular relaxations. Finally, the complete characterization of submodular functions up to degree $m = 4$ is instrumental to our work; see [1,27,34].

A preliminary version of this paper appeared in the conference proceedings of [16].

1.2. A brief example

As an example, consider the problem of minimizing the following cubic polynomial f over $\mathbf{B}^3$:

$$f(\boldsymbol{x}) = -2x_1 + x_2 - x_3 + 4x_1x_2 + 4x_1x_3 - 2x_2x_3 - 2x_1x_2x_3. \tag{3}$$

The standard reduction scheme [15] would use the identity $-x_1x_2x_3 = \min_{z\in\mathbf{B}} z(2 - x_1 - x_2 - x_3)$ to obtain a quadratic minimization problem with one auxiliary variable z. Roof duality gives a lower bound of $f_{\min} \geq -3$, but it does not reveal how to assign any of the variables in $\boldsymbol{x}$. However, there are many possible reduction schemes from which one can choose. Another possibility is $-x_1x_2x_3 = \min_{z\in\mathbf{B}} z(-x_1 + x_2 + x_3) - x_1x_2 - x_1x_3 + x_1$. For this reduction, the roof dual bound is tight and the optimal solution $\boldsymbol{x}^* = (0, 1, 1)$ is obtained (see Section 5). This simple example illustrates two facts: (i) different reductions lead to different lower bounds and (ii) it is not an obvious matter how to choose the optimal reduction.

1.3. Outline

In the next section, we introduce the concept of submodular relaxations and formulate the problem of finding relaxations that attain the maximum lower bound. Then, in Section 3, it is shown how to construct such relaxations in closed-form for quadratic pseudo-boolean functions. This construction turns out to be equivalent to the quadratic roof dual relaxation. For higher-order functions, things turn out to be more complicated. The generalized roof dual bound is analyzed in Section 4 and a polynomial-time algorithm is derived in order to compute the roof dual bound. In Sections 5 and 6, cubic and quartic relaxations are analyzed in more detail. A faster, but non-optimal heuristic method for constructing the relaxations is also proposed in Section 7. Experimental results on both synthetic and real data are presented in Section 8, and finally a concluding discussion is given.

2. Submodular relaxations

We will follow the framework of submodular (and bisubmodular) relaxations introduced in [18]. Consider the optimization problem in (2) where f has n variables. Without loss of generality, it is assumed that $f(\mathbf{0}) = 0$. By enlarging the domain, we will relax the problem and look at the following tractable problem:

$$\min_{(\boldsymbol{x},\boldsymbol{y})\in\mathbf{B}^{2n}} g(\boldsymbol{x},\boldsymbol{y}), \tag{4}$$

where $g\colon \mathbf{B}^{2n} \to \mathbf{R}$ is a pseudo-boolean function that satisfies the three conditions

$$g(\boldsymbol{x},\bar{\boldsymbol{x}}) = f(\boldsymbol{x}), \quad \forall \boldsymbol{x} \in \mathbf{B}^n, \tag{A}$$

$$g \text{ submodular}, \tag{B}$$

$$g(\boldsymbol{x},\boldsymbol{y}) = g(\bar{\boldsymbol{y}},\bar{\boldsymbol{x}}), \quad \forall (\boldsymbol{x},\boldsymbol{y}) \in \mathbf{B}^{2n} \text{ (symmetry)}. \tag{C}$$

For a point $\boldsymbol{x} = (x_1, x_2, \dots, x_n) \in \mathbf{B}^n$, we denote $\bar{\boldsymbol{x}} = (\bar{x}_1, \bar{x}_2, \dots, \bar{x}_n) = (1-x_1, 1-x_2, \dots, 1-x_n)$. The reason for requirement (A) is that if the range of f is included in the range of g then the minimum of g is a lower bound to the minimum of f. If the computed minimizer $(\boldsymbol{x}^*, \boldsymbol{y}^*)$ of the relaxation g happens to fulfill $\boldsymbol{x}^* = \bar{\boldsymbol{y}}^*$ then, of course, $\boldsymbol{x}^*$ is a minimizer of f as well. Even if it is not the case that $\boldsymbol{x}^* = \bar{\boldsymbol{y}}^*$, we still obtain a lower bound on f and as we shall see, it is possible to extract a partial solution for a minimizer of f.

Requirement (B) is also fairly obvious. Since we must be able to minimize g, requiring that g is submodular is natural. The last requirement will be motivated below.

2.1. Problem formulation

Let $f_{\min}$ denote the unknown minimum value of f, that is, $f_{\min} = \min f(\boldsymbol{x})$. Ideally, we would like $g(\boldsymbol{x},\boldsymbol{y}) \geq f_{\min}$ for all points $(\boldsymbol{x},\boldsymbol{y}) \in \mathbf{B}^{2n}$. This is evidently not possible in general. However, one could try to maximize the lower bound of g, $\max \min_{\boldsymbol{x},\boldsymbol{y}} g(\boldsymbol{x},\boldsymbol{y})$, that is,

$$\begin{array}{ll} \max\limits_{g,\ell} & \ell \\ \text{subject to} & g(\boldsymbol{x},\boldsymbol{y}) \geq \ell, \forall (\boldsymbol{x},\boldsymbol{y}) \in \mathbf{B}^{2n}, \\ & g \text{ satisfies (A)–(C)}. \end{array} \tag{5}$$

A relaxation g that provides the maximum lower bound will be called *optimal*. Note that the problem involves exponentially many constraints on g and therefore may seem like an intractable problem. As we shall prove in Section 3, when $m = 2$, the optimal relaxation can be constructed in closed-form. The lower bound coincides with the roof duality bound and therefore this bound will be referred to as *generalized roof duality* [18]. In Section 4, the general case will be analyzed and it will be shown how to compute the solution when the maximum of g is taken over a restricted set of submodular functions *in spite of* exponentially many constraints on g.

Symmetry. The last requirement (C), which specifies symmetry, is perhaps not so obvious. It is included for two reasons.

- We show below that restricting ourselves to this class of symmetric functions does not affect the obtained lower bound, i.e. the optimal relaxation is symmetric.
- For a symmetric g it is possible to prove *persistency*—which means that a partial solution can be extracted even though the complete, globally optimal solution is intractable.

A pseudo-boolean function g can be decomposed into a symmetric and an antisymmetric part, $g(\boldsymbol{x},\boldsymbol{y}) = g_{\text{sym}}(\boldsymbol{x},\boldsymbol{y}) + g_{\text{asym}}(\boldsymbol{x},\boldsymbol{y})$, where the symmetric part is defined by $g_{\text{sym}}(\boldsymbol{x},\boldsymbol{y}) = \frac{1}{2}(g(\boldsymbol{x},\boldsymbol{y}) + g(\bar{\boldsymbol{y}},\bar{\boldsymbol{x}}))$ and the antisymmetric part by $g_{\text{asym}}(\boldsymbol{x},\boldsymbol{y}) = \frac{1}{2}(g(\boldsymbol{x},\boldsymbol{y}) - g(\bar{\boldsymbol{y}},\bar{\boldsymbol{x}}))$. Note that $g_{\text{sym}}(\boldsymbol{x},\boldsymbol{y}) = g_{\text{sym}}(\bar{\boldsymbol{y}},\bar{\boldsymbol{x}})$ and $g_{\text{asym}}(\boldsymbol{x},\boldsymbol{y}) = -g_{\text{asym}}(\bar{\boldsymbol{y}},\bar{\boldsymbol{x}})$. If g satisfies requirements (A) and (B), then so does g_{sym}.

Consider the function g evaluated at the two points $(\boldsymbol{x},\boldsymbol{y})$ and $(\bar{\boldsymbol{y}},\bar{\boldsymbol{x}})$. We want the function values to be larger than some lower bound ℓ; hence $g(\boldsymbol{x},\boldsymbol{y}) = g_{\text{sym}}(\boldsymbol{x},\boldsymbol{y}) + g_{\text{asym}}(\boldsymbol{x},\boldsymbol{y}) \geq \ell$ and $g(\bar{\boldsymbol{y}},\bar{\boldsymbol{x}}) = g_{\text{sym}}(\boldsymbol{x},\boldsymbol{y}) - g_{\text{asym}}(\boldsymbol{x},\boldsymbol{y}) \geq \ell$. In order to achieve a maximum lower bound, it follows that $g_{\text{asym}}(\boldsymbol{x},\boldsymbol{y}) = 0$. Thus, to solve (5), restricting our attention to symmetric pseudo-boolean functions is enough.

Existence. The existence of feasible solutions for the optimization problem (5) can be seen from the following explicit example:

$$g(\boldsymbol{x},\boldsymbol{y}) = \begin{cases} f(\boldsymbol{x}) & \text{if } \boldsymbol{y} = \bar{x}, \\ -M \cdot \#\{i \mid x_i = y_i\} & \text{otherwise,} \end{cases} \tag{6}$$

where M is a sufficiently large constant. This gives, in some sense, the worst possible choice of g. Conditions (A) and (C) are satisfied by construction and submodularity (B) can be easily verified.

Linearity. Provided that the relaxation g is represented by a multilinear polynomial, constraint (A) is a linear equality constraint in the coefficients of g, as is constraint (C). The submodularity constraint can be expressed via linear inequality constraints; see Sections 5 and 6. Therefore, the optimization problem (5) is a linear program where the variables are the coefficients of g (and ℓ). As there always exists a feasible solution and the objective function is bounded from above, the concept of an optimal relaxation is well-defined.

Notation. As is standard, $\boldsymbol{x} \wedge \boldsymbol{y}$ and $\boldsymbol{x} \vee \boldsymbol{y}$ mean element-wise min and max, respectively. Let

$$\mathbf{S}^n = \{(\boldsymbol{x},\boldsymbol{y}) \in \mathbf{B}^{2n} \mid (x_i, y_i) \neq (1,1), i = 1,\ldots,n\}. \tag{7}$$

For $(x_1, y_1) \in \mathbf{S}^1$ and $(x_2, y_2) \in \mathbf{S}^1$, the operators $\sqcap$ and $\sqcup$ are defined by

$$\begin{aligned} (x_1, y_1) \sqcap (x_2, y_2) &= (x_1 \wedge x_2, y_1 \wedge y_2) \\ (x_1, y_1) \sqcup (x_2, y_2) &= \begin{cases} (0,0) & \text{if } (x_1 \vee x_2, y_1 \vee y_2) = (1,1) \\ (x_1 \vee x_2, y_1 \vee y_2) & \text{otherwise.} \end{cases} \end{aligned} \tag{8}$$

For $(\boldsymbol{x}_1, \boldsymbol{y}_1) \in \mathbf{S}^n$ and $(\boldsymbol{x}_2, \boldsymbol{y}_2) \in \mathbf{S}^n$, these operators are extended element-wise. Note that the resulting points still belong to $\mathbf{S}^n$. Further, for a scalar a, its positive and negative parts will be denoted by a^+ and a^-, where $a^+ = \max(a, 0)$ and $a^- = -\min(a, 0)$, respectively and hence $a = a^+ - a^-$. The conventions $a^+_{ij\bullet} = \sum_k a^+_{ijk}$ and $|a|_{ij\bullet\bullet} = \sum_{k<l} |a_{ijkl}|$ are also used for the ease of notation.

Relationship to bisubmodular functions. For any point $(\boldsymbol{x},\boldsymbol{y}) \in \mathbf{B}^{2n}$, it follows from the submodularity and symmetry of g that

$$g(\boldsymbol{x},\boldsymbol{y}) = \frac{1}{2}\left(g(\boldsymbol{x},\boldsymbol{y}) + g(\bar{\boldsymbol{y}},\bar{\boldsymbol{x}})\right) \geq \frac{1}{2}\left(g(\boldsymbol{x}\wedge\bar{\boldsymbol{y}},\boldsymbol{y}\wedge\bar{\boldsymbol{x}}) + g(\boldsymbol{x}\vee\bar{\boldsymbol{y}},\boldsymbol{y}\vee\bar{\boldsymbol{x}})\right) = g(\boldsymbol{x}\wedge\bar{\boldsymbol{y}},\boldsymbol{y}\wedge\bar{\boldsymbol{x}}),$$

where $(\boldsymbol{x}\wedge\bar{\boldsymbol{y}},\boldsymbol{y}\wedge\bar{\boldsymbol{x}}) \in \mathbf{S}^n$. So, when analyzing $\min_{(\boldsymbol{x},\boldsymbol{y})} g(\boldsymbol{x},\boldsymbol{y})$, considering the points in $\mathbf{S}^n$ is enough. Also, for any two points $(\boldsymbol{x}_1,\boldsymbol{y}_1) \in \mathbf{S}^n$ and $(\boldsymbol{x}_2,\boldsymbol{y}_2) \in \mathbf{S}^n$, we get

$$\begin{aligned} g(\boldsymbol{x}_1,\boldsymbol{y}_1) + g(\boldsymbol{x}_2,\boldsymbol{y}_2) &\geq g(\boldsymbol{x}_1\wedge\boldsymbol{x}_2,\boldsymbol{y}_1\wedge\boldsymbol{y}_2) + g(\boldsymbol{x}_1\vee\boldsymbol{x}_2,\boldsymbol{y}_1\vee\boldsymbol{y}_2) \\ &\geq g(\boldsymbol{x}_1\wedge\boldsymbol{x}_2,\boldsymbol{y}_1\wedge\boldsymbol{y}_2) + g((\boldsymbol{x}_1\vee\boldsymbol{x}_2)\wedge(\overline{\boldsymbol{y}_1\vee\boldsymbol{y}_2}),(\boldsymbol{y}_1\vee y_2)\wedge(\overline{\boldsymbol{x}_1\vee x_2})) \\ &= g((\boldsymbol{x}_1,\boldsymbol{y}_1)\sqcap(\boldsymbol{x}_2,\boldsymbol{y}_2)) + g((\boldsymbol{x}_1,\boldsymbol{y}_1)\sqcup(\boldsymbol{x}_2,\boldsymbol{y}_2)). \end{aligned} \tag{9}$$

By definition, a function satisfying

$$g(\boldsymbol{x}_1,\boldsymbol{y}_1) + g(\boldsymbol{x}_2,\boldsymbol{y}_2) \geq g((\boldsymbol{x}_1,\boldsymbol{y}_1)\sqcap(\boldsymbol{x}_2,\boldsymbol{y}_2)) + g((\boldsymbol{x}_1,\boldsymbol{y}_1)\sqcup(\boldsymbol{x}_2,\boldsymbol{y}_2))$$

is called *bisubmodular* [10] and hence, the restriction $g\colon \mathbf{S}^n \to \mathbf{R}$ is indeed a bisubmodular function. Note however that not all bisubmodular relaxations are submodular. The class of bisubmodular functions $\mathbf{S}^n \to \mathbf{R}$ is strictly larger than the class of submodular functions defined on the same domain.

2.2. Persistency

We start by defining the "overwrite" operator.

Definition 1. For any point $\boldsymbol{x} \in \mathbf{B}^n$ and $(\boldsymbol{x}^*,\boldsymbol{y}^*) \in \mathbf{S}^n$, the operator $\mathbf{B}^n \times \mathbf{S}^n \to \mathbf{B}^n$ denoted $\boldsymbol{x} \leftarrow (\boldsymbol{x}^*,\boldsymbol{y}^*)$ is defined by

$$\boldsymbol{x} \leftarrow (\boldsymbol{x}^*,\boldsymbol{y}^*) = \boldsymbol{u} \quad \text{where } (\boldsymbol{u},\bar{\boldsymbol{u}}) = ((\boldsymbol{x},\bar{\boldsymbol{x}}) \sqcup (\boldsymbol{x}^*,\boldsymbol{y}^*)) \sqcup (\boldsymbol{x}^*,\boldsymbol{y}^*). \tag{10}$$

One can check that this is well-defined and that

$$u_i = \begin{cases} x_i^*, & \text{if } (x_i^*, y_i^*) \neq (0,0) \\ x_i, & \text{otherwise} \end{cases} \quad \text{for } i = 1,\ldots,n. \tag{11}$$

So, $\boldsymbol{x} \leftarrow (\boldsymbol{x}^*,\boldsymbol{y}^*)$ can be thought of as the result of replacing elements of $\boldsymbol{x}$ by elements of $\boldsymbol{x}^*$ provided the corresponding element pairs in $(\boldsymbol{x}^*,\boldsymbol{y}^*)$ are non-zero.

Lemma 2 (*Autarky*). *Let g be a function satisfying (A)–(C) and $(\boldsymbol{x}^*, \boldsymbol{y}^*) \in \operatorname{argmin} g$. Then we have $f(\boldsymbol{x} \leftarrow (\boldsymbol{x}^*, \boldsymbol{y}^*)) \le f(\boldsymbol{x})$ for all $\boldsymbol{x}$.*

Proof. From bisubmodularity, it follows that, for any $\boldsymbol{v} \in \mathbf{B}^n$,

$$g((\boldsymbol{v}, \bar{\boldsymbol{v}}) \sqcup (\boldsymbol{x}^*, \boldsymbol{y}^*)) \le g(\boldsymbol{v}, \bar{\boldsymbol{v}}) + \left[g(\boldsymbol{x}^*, \boldsymbol{y}^*) - g((\boldsymbol{v}, \bar{\boldsymbol{v}}) \sqcap (\boldsymbol{x}^*, \boldsymbol{y}^*))\right] \le g(\boldsymbol{v}, \bar{\boldsymbol{v}}), \tag{12}$$

which implies that, for any $\boldsymbol{x} \in \mathbf{B}^n$,

$$f(\boldsymbol{x}) = g(\boldsymbol{x}, \bar{\boldsymbol{x}}) \ge g((\boldsymbol{x}, \bar{\boldsymbol{x}}) \sqcup (\boldsymbol{x}^*, \boldsymbol{y}^*)) \ge g(((\boldsymbol{x}, \bar{\boldsymbol{x}}) \sqcup (\boldsymbol{x}^*, \boldsymbol{y}^*)) \sqcup (\boldsymbol{x}^*, \boldsymbol{y}^*)) = f(\boldsymbol{x} \leftarrow (\boldsymbol{x}^*, \boldsymbol{y}^*)). \quad \square \tag{13}$$

This argument is due to [18]. Autarky is needed to ensure that the objective function does not increase when we use generalized roof duality in a move-making framework (see Section 8.2). An arguably more important consequence is persistency: if $\boldsymbol{x} \in \operatorname{argmin}(f)$, then $\boldsymbol{x} \leftarrow (\boldsymbol{x}^*, \boldsymbol{y}^*) \in \operatorname{argmin}(f)$. Hence, we have also proven the following special case.

Lemma 3 (*Persistency*). *Let g be a function satisfying (A)–(C) and $(\boldsymbol{x}^*, \boldsymbol{y}^*) \in \operatorname{argmin} g$. If $\boldsymbol{x} \in \operatorname{argmin}(f)$, then $\boldsymbol{x} \leftarrow (\boldsymbol{x}^*, \boldsymbol{y}^*) \in \operatorname{argmin}(f)$.*

In other words, all elements (x_i^*, y_i^*) not equal to $(0, 0)$ of a minimizer of g give us the corresponding elements x_i^* of a minimizer of f.

Remark 4. Lemmas 2 and 3 hold for *any* feasible relaxation g, not just the optimal one. This fact will be used later on.

3. Standard roof duality

We will start by analyzing quadratic submodular relaxations g of a quadratic pseudo-boolean function f. As we shall see, this is no restriction—optimal relaxations are of degree two.

A symmetric polynomial $g\colon \mathbf{B}^{2n} \to \mathbf{R}$ with $\operatorname{degree}(g) = 2$ can be represented by

$$g(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{2}\sum_i b_i(x_i + \bar{y}_i) + \sum_i b_{ii}x_i\bar{y}_i + \frac{1}{2}\sum_{i<j}\left(b_{ij}(x_ix_j + \bar{y}_i\bar{y}_j) + c_{ij}(x_i\bar{y}_j + \bar{y}_ix_j)\right). \tag{14}$$

The above expression contains all monomials of degree two or less and symmetry forces some of them to have equal coefficients. Thus (14) represents all quadratic and symmetric pseudo-boolean functions.

Lemma 5. *If the quadratic pseudo-boolean function f is represented by a multilinear polynomial (1) and the symmetric function g by (14), then the constraint $g(\boldsymbol{x}, \bar{\boldsymbol{x}}) = f(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathbf{B}^n$ implies that*

$$b_i + b_{ii} = a_i \quad \text{for } 1 \le i \le n \quad \text{and} \quad b_{ij} + c_{ij} = a_{ij} \text{ for } 1 \le i < j \le n. \tag{15}$$

Proof. For a given pseudo-boolean function, the multilinear polynomial representation (1) is unique; see [3]. So, it is enough to evaluate $g(\boldsymbol{x}, \bar{\boldsymbol{x}})$ and set the corresponding coefficients equal in the multilinear representations of $g(\boldsymbol{x}, \bar{\boldsymbol{x}})$ and $f(\boldsymbol{x})$. $\square$

It is well-known that a necessary and sufficient condition for g to be submodular is that the coefficients of the purely quadratic terms are non-positive in the multilinear representation. For a submodular, symmetric polynomial g in the form (14), this is equivalent to

$$b_{ii} \ge 0, \qquad b_{ij} \le 0 \quad \text{and} \quad c_{ij} \ge 0. \tag{16}$$

It follows that $b_{ij} = a_{ij} - c_{ij} \le a_{ij}$ and therefore $b_{ij} \le \min(a_{ij}, 0) = -a_{ij}^-$ and $c_{ij} \ge \max(a_{ij}, 0) = a_{ij}^+$.

The roof dual construction given in [3] proposes to set $b_{ij} = -a_{ij}^-$ and $c_{ij} = a_{ij}^+$, so it is in fact a quadratic submodular relaxation. We shall prove a stronger statement, namely that this relaxation g dominates any other bisubmodular relaxation $\tilde{g}$ of arbitrary degree, that is, $g(\boldsymbol{x}, \boldsymbol{y}) \ge \tilde{g}(\boldsymbol{x}, \boldsymbol{y})$ for all $(\boldsymbol{x}, \boldsymbol{y}) \in \mathbf{S}^n$. This result is already known, but we give a proof below using the notation of this paper.

Theorem 6 (*Kolmogorov [18]*). *An optimal submodular relaxation g of a quadratic pseudo-boolean function f is obtained through roof duality.*

1. *Set $b_i = a_i$ and $b_{ii} = 0$ for $1 \le i \le n$ in (14).*
2. *Set $b_{ij} = -a_{ij}^-$ and $c_{ij} = a_{ij}^+$ for $1 \le i < j \le n$ in (14).*

Further, the relaxation g is optimal among all possible bisubmodular relaxations $\tilde{g}$.

Proof. The proof is by induction over n.

For $n = 1$, we have $f(x_1) = g(x_1, \bar{x}_1) = \tilde{g}(x_1, \bar{x}_1)$. If $f(x_1) = a_1 x_1$ then $g(x_1, y_1) = \frac{1}{2}a_1(x_1 + \bar{y}_1)$ and

$$g(0,0) = \frac{1}{2}a_1 = \frac{1}{2}\left(\tilde{g}(0,1) + \tilde{g}(1,0)\right) \geq \tilde{g}(0,0),$$

which follows from symmetry and bisubmodularity of $\tilde{g}$.

For $n > 1$, assume that the statement holds for $n-1$ variables. Let $\boldsymbol{e}_i$ be a vector with zeros everywhere except at position i. Then, for any $i = 1, \ldots, n$, note that $g(\boldsymbol{x} \wedge \bar{\boldsymbol{e}}_i, \boldsymbol{y} \vee \boldsymbol{e}_i)$ is an optimal relaxation of $f(\boldsymbol{x} \wedge \bar{\boldsymbol{e}}_i)$ and hence

$$g(\boldsymbol{x} \wedge \bar{\boldsymbol{e}}_i, \boldsymbol{y} \vee \boldsymbol{e}_i) \geq \tilde{g}(\boldsymbol{x} \wedge \bar{\boldsymbol{e}}_i, \boldsymbol{y} \vee \boldsymbol{e}_i) \quad \text{for all } (\boldsymbol{x}, \boldsymbol{y}) \in \mathbf{S}^n.$$

In a similar manner,

$$g(\boldsymbol{x} \vee \boldsymbol{e}_i, \boldsymbol{y} \wedge \bar{\boldsymbol{e}}_i) \geq \tilde{g}(\boldsymbol{x} \vee \boldsymbol{e}_i, \boldsymbol{y} \wedge \bar{\boldsymbol{e}}_i) \quad \text{for all } (\boldsymbol{x}, \boldsymbol{y}) \in \mathbf{S}^n.$$

The only point not checked in $\mathbf{S}^n$ is $(\mathbf{0}, \mathbf{0})$. It suffices to show that $g(\mathbf{0}, \mathbf{0}) - g(\mathbf{0}, \boldsymbol{e}_1) \geq \tilde{g}(\mathbf{0}, \mathbf{0}) - \tilde{g}(\mathbf{0}, \boldsymbol{e}_1)$ since then it follows that

$$g(\mathbf{0}, \mathbf{0}) \geq \tilde{g}(\mathbf{0}, \mathbf{0}) + \underbrace{(g(\mathbf{0}, \boldsymbol{e}_1) - \tilde{g}(\mathbf{0}, \boldsymbol{e}_1))}_{\geq 0} \geq \tilde{g}(\mathbf{0}, \mathbf{0}).$$

Now, let $\boldsymbol{u} \in \mathbf{B}^n$ be defined by $u_1 = 0$ and

$$u_i = \begin{cases} 1, & \text{if } a_{1i} < 0 \\ 0, & \text{otherwise} \end{cases} \quad \text{for } i = 2, \ldots, n.$$

Then,

$$g(\mathbf{0}, \mathbf{0}) - g(\mathbf{0}, \boldsymbol{e}_1) = g(\boldsymbol{u}, \bar{\boldsymbol{u}} \wedge \bar{\boldsymbol{e}}_1) - g(\boldsymbol{u}, \bar{\boldsymbol{u}}) \geq \tilde{g}(\boldsymbol{u}, \bar{\boldsymbol{u}} \wedge \bar{\boldsymbol{e}}_1) - \tilde{g}(\boldsymbol{u}, \bar{\boldsymbol{u}}) \geq \tilde{g}(\mathbf{0}, \mathbf{0}) - \tilde{g}(\mathbf{0}, \boldsymbol{e}_1),$$

where the equality can be verified by straightforward calculations, and the first inequality is due to $g(\boldsymbol{u}, \bar{\boldsymbol{u}} \wedge \bar{\boldsymbol{e}}_1) \geq \tilde{g}(\boldsymbol{u}, \bar{\boldsymbol{u}} \wedge \bar{\boldsymbol{e}}_1)$ as already shown and the fact that $g(\boldsymbol{u}, \bar{\boldsymbol{u}}) = \tilde{g}(\boldsymbol{u}, \bar{\boldsymbol{u}})$. The last inequality is due to bisubmodularity. □

The roof dual bound is also known to be the tightest bound for several different linear programming relaxations [13].

4. Generalized roof duality

For a pseudo-boolean function f in n variables with degree$(f) > 2$, directly solving (5) is not tractable since the required number of constraints is exponential in n. Two obvious heuristic alternatives are the following.

1. Decompose f into a sum of the form $f(\boldsymbol{x}) = \sum_{i<j<k\cdots} f_{ijk\cdots}(x_i, x_j, x_k, \ldots)$ and compute an optimal relaxation for each term $f_{ijk\cdots}$. However, the sum of optimal relaxations is generally not optimal.
2. Use a subset of the points in $\mathbf{S}^n$ for $g(\boldsymbol{x}, \boldsymbol{y}) \geq \ell$ to get an approximate optimal relaxation.

Neither of these approaches are satisfactory. One may even wonder if the optimal relaxation g is polynomial time computable at all?

4.1. Definition

One important issue is to make sure that the set of submodular relaxations can be expressed in an easy manner, and in the end, be minimized by max-flow/min-cut. For this purpose, we adopt the notation of expressibility from [34].

Definition 7. A function $h: \mathbf{B}^n \to \mathbf{R}$ is called *expressible* if it can be expressed as $h(\boldsymbol{x}) = \min_{\boldsymbol{x}' \in \mathbf{B}^k} h'(\boldsymbol{x}, \boldsymbol{x}')$ for some k, where $h'(\boldsymbol{x}, \boldsymbol{x}')$ is a quadratic submodular function. Here $\boldsymbol{x}'$ are called *auxiliary variables*.

An expressible function is always submodular. From the definition of submodularity, it follows that a submodular function should satisfy exponentially many inequality constraints, but this is intractable. On the other hand, we have seen that for quadratic submodular functions, polynomially (or even linearly) many constraints are enough to define the set of submodular functions; see (16).

Definition 8. Consider a set of pseudo-boolean functions (up to a fixed degree) in n variables parametrized by a coefficient vector $\boldsymbol{a} \in \mathbf{R}^d$. A subset of submodular functions is called *recognizable* if the submodularity condition can be expressed by polynomially many linear inequality constraints in $\boldsymbol{a}$ with respect to n.

In Section 5 we will reiterate a well-known fact: all cubic submodular functions are expressible, and the set of cubic submodular functions is recognizable among the set of cubic functions. Unfortunately, all quartic submodular functions are not expressible [34] and whether the subset of expressible functions is recognizable is an open problem. This makes the quartic and higher-order generalization of the roof dual much harder to handle than the quadratic and cubic cases. In Section 6 we will define recognizable sets of quartic expressible functions in two different ways, and investigate their properties.

Henceforth, requirement (B) is replaced by the following extended condition,

$$g \in \mathcal{G}, \quad \text{where } \mathcal{G} \text{ is a recognizable set of expressible functions.} \tag{B$'$}$$

The precise definition of roof duality is then as follows.

Definition 9 (*Generalized Roof Duality*)**.** The *generalized roof duality* bound over a recognizable set of expressible functions is the optimal value of (5) with constraints (A), (B$'$) and (C). The optimal value will be denoted by g^*_{GRD}.

Note that computing the optimal value g^*_{GRD} directly via (5) still involves exponentially many constraints due to $g(\boldsymbol{x}, \boldsymbol{y}) \geq \ell$ for all $(\boldsymbol{x}, \boldsymbol{y}) \in \mathbf{S}^n$.

4.2. Main result

Lemma 3 states that persistency holds for any bisubmodular relaxation g—optimal or not. From the example in (3), it is clear, however, that not all relaxations are equally powerful. Instead of solving (5), which, although possible, may require a large number of constraints, one can consider a simpler problem:

$$\begin{aligned} \max_{g} \quad & g(\mathbf{0}, \mathbf{0}) \\ \text{subject to} \quad & g \text{ satisfies (A), (B}'\text{) and (C).} \end{aligned} \tag{17}$$

Instead of maximizing $\min g(\boldsymbol{x}, \boldsymbol{y})$, we are only maximizing $g(\mathbf{0}, \mathbf{0})$. This problem is considerably less arduous and can be solved in polynomial time.[1] Given the minimizer $(\boldsymbol{x}^*, \boldsymbol{y}^*) \in \operatorname{argmin}(g)$, which is also polynomial time computable since g is submodular, we can make the following important observations.

- If $(\boldsymbol{x}^*, \boldsymbol{y}^*)$ is non-zero, then we can use persistency to get a partial solution and reduce the number of variables in f.
- Otherwise, as the optimum is indeed the trivial solution, and as $g(\mathbf{0}, \mathbf{0})$ is maximized in the construction of g, then we can conclude that g is an optimal relaxation and we have obtained the generalized roof duality bound.

These observations lead to the following algorithm that computes the generalized roof duality bound.

1. Construct g by solving (17).
2. Compute $(\boldsymbol{x}^*, \boldsymbol{y}^*) \in \operatorname{argmin}(g)$.
3. If $(\boldsymbol{x}^*, \boldsymbol{y}^*)$ is non-zero, use persistency to simplify f and start over from 1. Otherwise, stop.

Theorem 10. *A lower bound on* $\min f(\boldsymbol{x})$ *which is greater than or equal to the generalized roof duality bound* g^*_{GRD} *over a recognizable set of expressible pseudo-boolean functions can be computed in polynomial time.*

Proof. If $(\boldsymbol{x}^*, \boldsymbol{y}^*)$ is non-zero, persistency can be used to simplify the original function f. This is equivalent to adding constraints of the type $x_i = \bar{y}_i = c$ to $\max_g \min_{\boldsymbol{x},\boldsymbol{y}} g(\boldsymbol{x}, \boldsymbol{y})$. This can only increase the computed value. If on the other hand $(\boldsymbol{x}^*, \boldsymbol{y}^*) = (\mathbf{0}, \mathbf{0})$, then the best possible lower bound is obtained by construction of g. Therefore, the final bound is at least equal to the optimal value g^*_{GRD} of (5).

The algorithm can obviously not run for more than n iterations, since in each iteration either persistencies are found and f is simplified or the algorithm terminates. With all steps being solvable in polynomial time, the algorithm itself is polynomial. □

For the cubic case $m = 3$, the theorem can be simplified.

Corollary 11. *A lower bound on* $\min f(\boldsymbol{x})$ *which is greater than or equal to the generalized roof duality bound* g^*_{GRD} *over cubic submodular functions can be computed in polynomial time.*

Remark 12. Note that we do not explicitly construct the optimal relaxation g, but rather a sequence of relaxations $g_1, g_2, \ldots, g_k$, such that the final relaxation g_k fulfills $\min_{\boldsymbol{x},\boldsymbol{y}} g_k(\boldsymbol{x}, \boldsymbol{y}) \geq g^*_{\mathrm{GRD}}$.

Remark 13. As suggested by the proof of Theorem 10, the iterative approach can obtain a better bound than the "optimal" value g^*_{GRD} in Definition 9. We have observed this in practice for small problems where directly solving (5) is feasible. One example is

$$\begin{aligned} f(\boldsymbol{x}) = {} & 14x_1 + 15x_2 - 6x_3 + 9x_4 - 5x_1x_2 + 6x_1x_3 + 3x_1x_4 + 13x_2x_3 + 13x_2x_4 - 6x_3x_4 \\ & + 20x_1x_2x_3 + 9x_1x_2x_4 + 17x_1x_3x_4 + 2x_2x_3x_4, \end{aligned} \tag{18}$$

for which (5) gives the lower bound $g^*_{\mathrm{GRD}} = -8$ and the iterative method above gives $-6 > g^*_{\mathrm{GRD}}$, which is tight.

---

[1] The condition $g(\boldsymbol{x}, \bar{\boldsymbol{x}}) = f(\boldsymbol{x})$ for all $\boldsymbol{x}$ does involve exponentially many constraints, but as g is required to be of fixed degree (independent of n), then its polynomial representation has only polynomially many terms.

5. Cubic relaxations

In this section, we will in detail analyze the properties of relaxations of degree three with respect to symmetry and submodularity.

A cubic symmetric polynomial $g\colon \mathbf{B}^{2n} \to \mathbf{R}$ can be written as

$$g(\boldsymbol{x}, \boldsymbol{y}) = L + Q + \frac{1}{2}\sum_{i<j}\left(b_{iij}x_i\bar{y}_i(x_j+\bar{y}_j) + b_{ijj}(x_i+\bar{y}_i)x_j\bar{y}_j\right) + \frac{1}{2}\sum_{i<j<k}\left(b_{ijk}(x_ix_jx_k + \bar{y}_i\bar{y}_j\bar{y}_k)\right.$$
$$\left. +\ c_{ijk}(x_ix_j\bar{y}_k + \bar{y}_i\bar{y}_jx_k) + d_{ijk}(x_i\bar{y}_jx_k + \bar{y}_ix_j\bar{y}_k) + e_{ijk}(\bar{y}_ix_jx_k + x_i\bar{y}_j\bar{y}_k)\right), \tag{19}$$

where L and Q denote linear and quadratic terms as in Section 3. The objective function in (17) is then simply equal to

$$g(\mathbf{0}, \mathbf{0}) = \frac{1}{2}\left(\sum_i b_i + \sum_{i<j} b_{ij} + \sum_{i<j<k} b_{ijk}\right). \tag{20}$$

Lemma 14. *If the cubic pseudo-boolean function f is represented by a multilinear polynomial (1) and the symmetric function g by (19), then the constraint $g(\boldsymbol{x}, \bar{\boldsymbol{x}}) = f(\boldsymbol{x})$ for all $\boldsymbol{x} \in \mathbf{B}^n$ implies that*

$$\begin{aligned} &b_i + b_{ii} = a_i \quad \text{for } 1 \le i \le n \\ &b_{ij} + c_{ij} + b_{iij} + b_{ijj} = a_{ij} \quad \text{for } 1 \le i < j \le n \\ &b_{ijk} + c_{ijk} + d_{ijk} + e_{ijk} = a_{ijk} \quad \text{for } 1 \le i < j < k \le n. \end{aligned} \tag{21}$$

Proof. See the proof of Lemma 5. □

The necessary and sufficient conditions for a cubic polynomial to be submodular were given in [1]. The characterization is slightly more complicated than the quadratic case: a cubic multilinear polynomial f in (1) is submodular if and only if, for every $i < j$,

$$a_{ij} + a^+_{ij\bullet} + a^+_{i\bullet j} + a^+_{\bullet ij} \le 0. \tag{22}$$

Here we give a new formulation suitable for our purposes. The set of symmetric, submodular functions of degree 3 will be denoted by $\Gamma_{\mathrm{sym},3}$.

Lemma 15. *A cubic symmetric polynomial g represented by (19) is submodular if and only if*

$$b_{ij} + b^+_{iij} + b^+_{ijj} \le -b^+_{ij\bullet} - b^+_{i\bullet j} - b^+_{\bullet ij} - c^+_{ij\bullet} - d^+_{i\bullet j} - e^+_{\bullet ij} \quad \text{for } 1 \le i < j \le n \tag{23a}$$

$$-c_{ij} + b^-_{iij} + b^-_{ijj} \le -c^-_{i\bullet j} - c^-_{\bullet ij} - d^-_{ij\bullet} - d^-_{\bullet ij} - e^-_{ij\bullet} - e^-_{i\bullet j} \quad \text{for } 1 \le i < j \le n \tag{23b}$$

$$b_{ii} \ge b^-_{ii\bullet} + b^-_{\bullet ii} \quad \text{for } 1 \le i \le n. \tag{23c}$$

Proof. Consider the multilinear representation of g. If we apply condition (22) for terms involving the monomial x_ix_j, then the linear inequality (23a) is obtained. Exactly the same inequality is obtained for terms involving the monomial y_iy_j (by symmetry). Further applications of (22) for the monomial x_iy_j (and its twin y_ix_j) result in (23b). Finally, terms involving x_iy_i give rise to condition (23c). As there are no other quadratic monomials appearing in g, the proof is complete. □

Lemma 16. *There is a solution g represented by (19) to the optimization problem (17) such that*

(i) *$b_{ii} = 0$ for $1 \le i \le n$, $b_{iij} = b_{ijj} = 0$ for $1 \le i < j \le n$,*
(ii) *if $a_{ijk} \ge 0$ then $b^-_{ijk} = c^-_{ijk} = d^-_{ijk} = e^-_{ijk} = 0$ for $1 \le i < j < k \le n$,*
(iii) *if $a_{ijk} \le 0$ then $b^+_{ijk} = c^+_{ijk} = d^+_{ijk} = e^+_{ijk} = 0$ for $1 \le i < j < k \le n$.*

Proof. (i) Suppose that $b_{iij} > 0$ for the optimal g. Then we see from (21) and (23a) that setting b_{iij} to 0 and increasing b_{ij} by the same amount will still be feasible. This operation increases the objective function $g(\mathbf{0}, \mathbf{0})$. If, on the other hand, $b_{iij} < 0$, then b_{iij} can also be set to 0 and decreasing c_{ij} by the same amount will give a feasible solution, see (21) and (23b), with no change to the objective function. The same argument holds for b_{ijj}. Finally, setting $b_{ii} = 0$ and increasing b_i by the same amount will always be feasible and increase the objective function.

(ii) Suppose $b^-_{ijk} > 0$. Then setting $b^-_{ijk} = 0$ and decreasing c^+_{ijk}, d^+_{ijk} and e^+_{ijk} such that the sum $c^+_{ijk} + d^+_{ijk} + e^+_{ijk}$ is decreased by the same amount will still be a feasible solution with higher objective function value. Other variables are similarly handled.

(iii) The proof is analogous to (ii). □

The above lemma simplifies matters. If, say $a_{ijk} > 0$, then we can set $b_{ijk} = b^+_{ijk}$, $c_{ijk} = c^+_{ijk}$, $d_{ijk} = d^+_{ijk}$ and $e_{ijk} = e^+_{ijk}$. Further, the submodularity conditions (23a) and (23b) become linear inequality constraints in the unknowns, condition (23c) becomes obsolete and the optimization problem (17) is turned into an instance of linear programming.

Example 17. Consider again the example of a cubic pseudo-boolean function f in (3). Finding a $g(\boldsymbol{x}, \boldsymbol{y})$ of the form (19) by solving (17) results in

$$g(\boldsymbol{x}, \boldsymbol{y}) = -(x_1 + \bar{y}_1) + \frac{1}{2}(x_2 + \bar{y}_2) - \frac{1}{2}(x_3 + \bar{y}_3) + 2(x_1\bar{y}_2 + \bar{y}_1x_2) + 2(x_1\bar{y}_3 + \bar{y}_1x_3) \\ - (x_2x_3 + \bar{y}_2\bar{y}_3) - (\bar{y}_1x_2x_3 + x_1\bar{y}_2\bar{y}_3). \tag{24}$$

Minimizing this submodular relaxation gives $g_{\min} = -2$ for $(\boldsymbol{x}^*, \boldsymbol{y}^*) = (0, 1, 1, 1, 0, 0)$. Since $\boldsymbol{x}^* = \bar{\boldsymbol{y}}^*$, it follows that $\boldsymbol{x}^*$ is the global minimizer for f as well.

6. Quartic relaxations

Determining whether a given quartic polynomial is submodular or not is known to be co-NP-complete [6], and not all submodular quartic polynomials are expressible by quadratic submodular functions [34]. Therefore, a compromise is required. In this section, we define and analyze two different proposals of recognizable sets for quartic functions.

6.1. Approach I: the set $\Gamma_{\mathrm{sym},4}$

We choose to work with the quartic polynomials in (1) that satisfy, for every $i < j$,

$$a_{ij} + a^+_{ij\bullet} + a^+_{i\bullet j} + a^+_{\bullet ij} + a^+_{ij\bullet\bullet} + a^+_{i\bullet j\bullet} + \cdots + a^+_{\bullet\bullet ij} \leq 0. \tag{25}$$

This choice can be seen as a natural generalization of the cubic case; see (22). The set which we denote $\Gamma_{\mathrm{suff},4}$ has a number of advantageous properties. First, it is a rich set of submodular functions. For example, the set of cubic submodular functions, denoted $\Gamma_{\mathrm{suff},3}$, is a subset of these functions, $\Gamma_{\mathrm{suff},3} \subset \Gamma_{\mathrm{suff},4}$. See also [35] where the set is analyzed in more detail. Second, each quartic term only needs one auxiliary variable for expressibility. Finally, only $O(n^2)$ inequalities are sufficient to make sure that our relaxation is submodular. Thus, the set is recognizable among all quartic pseudo-boolean functions.

Similar to previous derivations, a quartic symmetric polynomial can be written as

$$g(\boldsymbol{x}, \boldsymbol{y}) = L + Q + C + \frac{1}{2}\sum_{i<j<k<l} \big(b_{ijkl}(x_ix_jx_kx_l + \bar{y}_i\bar{y}_j\bar{y}_k\bar{y}_l) + c_{ijkl}(x_ix_jx_k\bar{y}_l + \bar{y}_i\bar{y}_j\bar{y}_kx_l) \\ + d_{ijkl}(x_ix_j\bar{y}_kx_l + \bar{y}_i\bar{y}_jx_k\bar{y}_l) + e_{ijkl}(x_i\bar{y}_jx_kx_l + \bar{y}_ix_j\bar{y}_k\bar{y}_l) + p_{ijkl}(\bar{y}_ix_jx_kx_l + x_i\bar{y}_j\bar{y}_k\bar{y}_l) \\ + q_{ijkl}(x_ix_j\bar{y}_k\bar{y}_l + \bar{y}_i\bar{y}_jx_kx_l) + r_{ijkl}(x_i\bar{y}_jx_k\bar{y}_l + \bar{y}_ix_j\bar{y}_kx_l) + s_{ijkl}(x_i\bar{y}_j\bar{y}_kx_l + \bar{y}_ix_jx_k\bar{y}_l)\big), \tag{26}$$

where L, Q and C denote lower-order terms, and $b_{ijkl} + c_{ijkl} + \cdots + s_{ijkl} = a_{ijkl}$. We make analogous simplifications as for the cubic case, see Lemma 16, so it is assumed $b_{ii} = b_{iij} = b_{ijj} = 0$. The corresponding conditions for a symmetric quartic polynomial of this form to be in $\Gamma_{\mathrm{suff},4}$ are given in Lemma 19 in Appendix. We will denote the subset of symmetric functions by $\Gamma_{\mathrm{sym},4}$, and naturally we have $\Gamma_{\mathrm{sym},3} \subset \Gamma_{\mathrm{sym},4}$.

Example 18. In [15], the following reduction identity is proposed

$$x_1x_2x_3x_4 = \min_{z\in\mathbf{B}} z(3 - 2x_1 - 2x_2 - 2x_3 - 2x_4) + x_1x_2 + x_1x_3 + x_1x_4 + x_2x_3 + x_2x_4 + x_3x_4. \tag{27}$$

This can be used to express

$$f(\boldsymbol{x}) = x_1 + x_3 - x_4 + 2x_1x_4 + 2x_2x_3 - x_3x_4 + x_1x_2x_3x_4 \tag{28}$$

as a quadratic polynomial with one auxiliary variable z. The quadratic roof duality bound gives $f_{\min} \geq -2$ and no partial assignments. On the other hand, solving the linear program (17), one obtains the relaxation

$$g(\boldsymbol{x}, \boldsymbol{y}) = \frac{1}{2}(x_1 + \bar{y}_1) + \frac{1}{2}(x_3 + \bar{y}_3) - \frac{1}{2}(x_4 + \bar{y}_4) - \frac{1}{2}(x_1x_3 + \bar{y}_1\bar{y}_3) + \frac{1}{2}(x_1\bar{y}_3 + \bar{y}_1x_3) - \frac{1}{2}(x_1x_4 + \bar{y}_1\bar{y}_4) \\ + \frac{3}{2}(x_1\bar{y}_4 + \bar{y}_1x_4) + (x_2\bar{y}_3 + \bar{y}_2x_3) - \frac{1}{2}(x_3x_4 + \bar{y}_3\bar{y}_4) + \frac{1}{2}(x_1\bar{y}_2x_3x_4 + \bar{y}_1x_2\bar{y}_3\bar{y}_4).$$

Solving the submodular problem $\min g(\boldsymbol{x}, \boldsymbol{y})$ via max-flow yields $(\boldsymbol{x}^*, \boldsymbol{y}^*) = (0, 0, 0, 1, 1, 1, 1, 0)$. Again, since $\boldsymbol{x}^* = \bar{\boldsymbol{y}}^*$, it follows that $\boldsymbol{x}^*$ is the global minimizer for f, that is, $f_{\min} = g^*_{\mathrm{GRD}} = -1$.

6.2. Approach II: generators of expressible functions

Submodular pseudo-boolean functions form a convex cone in $\mathbf{R}^d$ [27]. Recall that a cone in a vector space is a set $\mathcal{C}$ such that $\mathbf{0} \in \mathcal{C}$ and $\lambda\boldsymbol{x} \in \mathcal{C}$ for every $\lambda \geq 0$ and every $\boldsymbol{x} \in \mathcal{C}$.

One way to work with a cone of expressible pseudo-boolean functions is to find a finite set of generators for the cone, that is, a set of pseudo-boolean functions $\{e_1, \ldots, e_k\}$ such that every function f in the cone can be written as $f(\boldsymbol{x}) = \sum_{i=1}^{k} \alpha_i e_i(\boldsymbol{x})$ for $\alpha_i \geq 0, i = 1, \ldots, k$. For $n = 4$ variables, generators of the submodular cone have been derived [27]. Apart from the linear functions, there are 10 generators (1 quadratic, 2 cubic and 7 quartic generators), see Fig. 2 in [36] for a complete list. Out of these 10 generators, one (quartic) generator is *not* expressible. This gives a convenient way to represent all expressible functions in 4 variables. We are of course interested in the cone of symmetric, expressible functions in $2n$ variables. However, it is an open problem to determine this cone's set of generators and even if the generators were known, working with the full set of generators is likely to be intractable.

Based on the generators $\{e_1, \ldots, e_k\}$ for the expressible cone of 4 variables, we will explicitly define a cone of symmetric and expressible functions over $\mathbf{S}^n$ via generators as follows. For every non-zero quartic coefficient a_{ijkl} of $f(\boldsymbol{x})$, and every quartic generator e_s where $s = 1, \ldots, k$, we construct the symmetric generator

$$e_s(x_i, x_j, x_k, x_l) + e_s(\bar{y}_i, \bar{y}_j, \bar{y}_k, \bar{y}_l).$$

Such a generator will not be able to generate functions with monomials consisting of both $\boldsymbol{x}$ and $\boldsymbol{y}$ variables. Therefore, we also construct generators by exchanging x_i and y_i

$$e_s(y_i, x_j, x_k, x_l) + e_s(\bar{x}_i, \bar{y}_j, \bar{y}_k, \bar{y}_l),$$

and similarly for the other variables. There are up to $2^4/2 = 8$ (and not 16 due to symmetry) such combinations for every e_s. In an analogous manner, quadratic and cubic generators are constructed for each pair and triplet of indices, respectively. This procedure creates, not counting duplicates:

- 2 quadratic generators for every combination i, j,

$$-x_i x_j - \bar{y}_i \bar{y}_j \quad \text{and} \quad -x_i y_j - \bar{y}_i \bar{x}_j.$$

- 8 cubic generators for every combination i, j, k,

$$-x_i x_j x_k - \bar{y}_i \bar{y}_j \bar{y}_k, -y_i x_j x_k - \bar{x}_i \bar{y}_j \bar{y}_k, \ldots, -x_i y_j y_k - \bar{y}_i \bar{x}_j \bar{x}_k, -y_i y_j y_k - \bar{x}_i \bar{x}_j \bar{x}_k.$$

- 132 quartic generators for every combination i, j, k, l, for example,

$$-x_i x_j x_k x_l - \bar{y}_i \bar{y}_j \bar{y}_k \bar{y}_l.$$

It can be shown that (i) the set of all submodular and symmetric cubic functions $\Gamma_{\text{sym},3}$ is generated by the quadratic and cubic generators above (modulo linear terms), and (ii) the set of symmetric quartic polynomials $\Gamma_{\text{sym},4}$ is a subcone of the cone generated by the generators above (modulo linear terms).

In order to compute the generalized roof dual bound, we need to be able to solve problem (17). Given $f(\boldsymbol{x})$ and the set of generators $\{e_1(\boldsymbol{x}, \boldsymbol{y}), \ldots, e_K(\boldsymbol{x}, \boldsymbol{y})\}$ as described above, we can express our submodular relaxation g as $g(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{K} \alpha_i e_i(\boldsymbol{x}, \boldsymbol{y})$. Constraint (A), that is, $g(\boldsymbol{x}, \bar{\boldsymbol{x}}) = f(\boldsymbol{x})$ can be written as $\boldsymbol{A}\boldsymbol{\alpha} = \boldsymbol{a}$, where $\boldsymbol{\alpha}$ is a vector of length K and $\boldsymbol{a}$ is a vector with all polynomial coefficients of f. Constraints (B$'$) and (C) are automatically satisfied as g is ensured to lie in a cone of expressible and symmetric functions. The objective function $g(0, 0)$ can be written as $\boldsymbol{c}^T\boldsymbol{\alpha}$ where $\boldsymbol{c}$ is a K-vector. In summary, the maximization problem in (17) using generators can be cast as a linear programming problem,

$$\begin{array}{ll} \max & \boldsymbol{c}^T\boldsymbol{\alpha} \\ \text{subject to} & \boldsymbol{A}\boldsymbol{\alpha} = \boldsymbol{a} \\ & \boldsymbol{\alpha} \geq 0. \end{array} \tag{29}$$

7. Heuristics

In many cases, the optimization problem (17) does not need to be solved exactly. Minimizing g amounts to solving a maximum flow problem, which is considerably faster than solving a linear program to create g. Simpler, heuristic methods which approximately maximize $g(\mathbf{0}, \mathbf{0})$ are therefore of interest.

It can be seen from Lemmas 15 and 19 that coefficients b_{ij} (and therefore c_{ij}) are completely determined if all coefficients of higher order are fixed. Since b_{ij} appears in the objective function, taking the minimum of (23a) and (23b) (and similarly in Lemma 19 for the quartic case) will give the optimal value.

Naturally, any heuristic can be combined with the generalized roof duality method. The following procedure can be used to compute the roof dual bound.

1. Use heuristics or any type of relaxations to obtain persistencies and simplify f.
2. Apply the generalized roof duality procedure from Section 4.

The end result will still attain the generalized roof dual bound g^*_{GRD} for the original function f, but much faster for some problems due to the fact that much smaller linear programs are solved.

Table 1
Abbreviations used in the experimental section.

GRD	Generalized Roof Duality (GRD) using $\Gamma_{\mathrm{sym},m}$ (Sections 5 and 6.1)
GRD-gen	GRD using generators for $m = 4$ (Section 6.2)
GRD-heuristic	The heuristic relaxations (Section 7)
Fix et al.	The reductions proposed in [8]
HOCR	The reductions proposed in [15]

7.1. Cubic case

For the cubic case we use the following heuristics: let f be written as the sum of functions $f_1 + f_2 + \cdots + f_N$, where each f_i is a function of three variables only. Computing the optimal g_i for each of these functions is possible to do very quickly. The sum of optimal relaxations is in general not optimal, as noted before, but the approximation might give a reasonable heuristic.

7.2. Quartic case

For the quartic case, we try an even simpler heuristic. We simply use the procedure from the cubic case and set

$$d_{ijkl} = a^+_{ijkl} \quad \text{and} \quad b_{ijkl} = -a^-_{ijkl}. \tag{30}$$

Even this simple method performs surprisingly well for some application problems, as we will show in the experimental section (see Figs. 2 and 3). Presumably this is due to the fact that setting b_{ij} to the minimum of (23a) and (23b) is optimal given that the higher-order coefficients are determined.

8. Experiments

In this section, we evaluate generalized roof duality experimentally. When computing the generalized roof duality, we used linear programming in every step, that is, we did not use any combination of linear programming and heuristics, as mentioned in Section 7. The exception is Section 8.2.2, where the problems were preprocessed with heuristic relaxations and simplified, after which the linear programming relaxation was computed. In practice, one would typically always use a combination of heuristics and linear programming, or heuristics only. Table 1 lists the abbreviations of the different relaxation methods.

8.1. Random polynomials

In the first experiment, we apply our method to synthetically generated polynomials with random coefficients:

$$f(\boldsymbol{x}) = \sum_{(i,j,k)\in T} f_{ijk}(x_i, x_j, x_k), \tag{31}$$

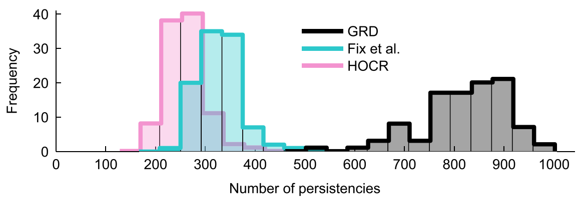
where $T \subseteq \{1 \cdots n\}^3$ is a random set of triplets and each f_{ijk} is a cubic polynomial in x_i, x_j and x_k with all its coefficients picked uniformly in $\{-100, \ldots, 100\}$. We minimize f with the different methods listed in Table 1. After each algorithm finishes, we count the number of persistencies (also called the number of labeled variables). The results from 100 problem instances can be seen in Fig. 1(a). For this type of polynomials, our submodular relaxations significantly outperform the previous state of the art for every problem. The time required to solve the linear program (17) was longer,[2] but in combination with heuristics this time may be shortened significantly. The minimum and maximum number of iterations required were 3 and 12, respectively, with 93% of the problem instances requiring 6 or less. In addition to comparing the number of persistencies, we also compared the achieved lower bounds by computing the relative difference: $(\ell_{\mathrm{GRD}} - \ell)/|\ell_{\mathrm{GRD}}|$.

We also generated random quartic polynomials in the same manner; see Fig. 1(b). This experiment also resulted in a large separation, and the relative lower bound differences were much larger. The best performing method in terms of lower bounds is, not surprisingly, the GRD method based on generators. Perhaps somewhat surprisingly, the GRD generator method is faster than the GRD method based on $\Gamma_{\mathrm{sym},4}$. Even though the linear program for the generator method is much bigger, the computations are faster. Note that we only used $|T| = 300$ for this experiment; with $|T| = 1000$ the generalized roof duality only obtained a median of 14 persistencies while HOCR obtained 3.

8.2. Applications in computer vision

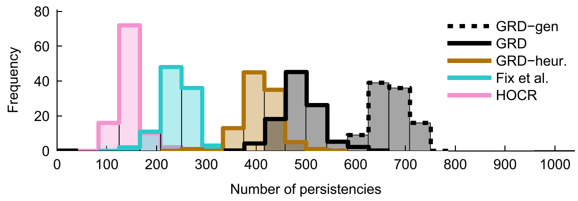
The minimization of pseudo-boolean functions appears in many different fields. Our original motivation behind this work stems from inference problems in computer vision and machine learning. For many low-level problems in computer vision, a Markov Random Field is used for modeling and the resulting inference problem consists of estimating some unknown

[2] We used Clp (http://www.coin-or.org/Clp) as our LP solver.



	Rel. bounds			Time (ms)		
	Min	Med.	Max	Min.	Med.	Max
GRD	0	0	0	422	514	749
GRD-heur.	0.00	0.00	0.00	31	63	125
Fix et al.	0.05	0.09	0.13	15	16	32
HOCR	0.10	0.14	0.20	0	15	31

(a) Cubic polynomials with $n = 1000$ and $|T| = 1000$. GRD-heuristic is not shown in the histogram because it is almost indistinguishable from GRD.



	Rel. bounds			Time (ms)		
	Min	Med.	Max	Min.	Med.	Max
GRD-gen.	−0.10	−0.07	−0.05	360	438	560
GRD	0	0	0	599	810	1478
GRD-heur.	0.04	0.06	0.09	60	100	160
Fix et al.	0.13	0.16	0.20	0	10	20
HOCR	0.42	0.48	0.55	0	10	17

(b) Quartic polynomials with $n = 1000$ and $|T| = 300$.

Fig. 1. Number of persistencies, relative bounds and running time for 100 random polynomials. The set of coefficients T was drawn uniformly after making sure that all variables were used once.

quantity (e.g., depths) from one or several observations. The objective function for such a problem (often referred to as an *energy*) to be minimized is typically of the form

$$E(\boldsymbol{w}) = \underbrace{\sum_{i=1}^{n} E_i(w_i)}_{E_{\text{data}}(\boldsymbol{w})} + \underbrace{\sum_{i<j} E_{ij}(w_i, w_j) + \sum_{i<j<k} E_{ijk}(w_i, w_j, w_k) + \cdots}_{E_{\text{smooth}}(\boldsymbol{w})}, \tag{32}$$

where the *data term* E_{data} specifies the agreement between $\boldsymbol{w}$ and the observations. The *smoothness term* E_{smooth} measures how well $\boldsymbol{w}$ agrees with prior information such as smoothness and noise levels in the image.

In many applications, $\boldsymbol{w}$ is not a boolean vector but instead $\boldsymbol{w} \in \{1, \dots, K\}^n$, where K is the number of labels. In practice, the multilabel problem is reduced to the boolean case by iterative move-making algorithms [5,23]: given a current solution $\boldsymbol{w}^{(t)}$ and a *proposal* $\boldsymbol{y}^{(t)}$, a new (better) solution is constructed from a solution $\boldsymbol{x}^*$ to a pseudo-boolean minimization problem,

$$w_i^{(t+1)} = \begin{cases} y_i^{(t)} & \text{if } x_i^* = 1 \\ w_i^{(t)} & \text{if } x_i^* = 0 \text{ (or unknown)}. \end{cases} \tag{33}$$

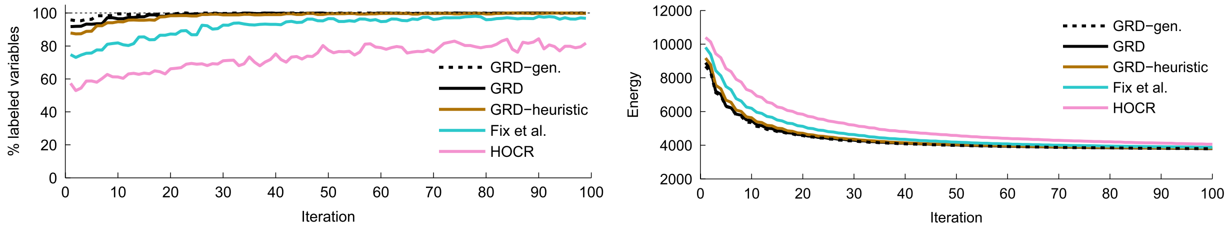
The pseudo-boolean objective function $E^{(t)}(\boldsymbol{x})$ at iteration t is constructed as follows. The data terms are given by $E_i^{(t)}(x_i) = E_i(y_i^{(t)})x_i + E_i(w_i^{(t)})\bar{x}_i$. Similarly, $E_{ij}^{(t)}(x_i, x_j) = E_{ij}(y_i^{(t)}, y_j^{(t)})x_ix_j + E_{ij}(y_i^{(t)}, w_j^{(t)})x_i\bar{x}_j + E_{ij}(w_i^{(t)}, y_j^{(t)})\bar{x}_ix_j + E_{ij}(w_i^{(t)}, w_j^{(t)})\bar{x}_i\bar{x}_j$ and so forth. Because of this, we refer the maximum number of variables appearing in each term of E_{smooth} as its degree. We will for the remainder of this subsection focus on two specific applications: image denoising, where the degree of E_{smooth} is 4, and stereo reconstruction, where the degree of E_{smooth} is 3.

8.2.1. *Image denoising*

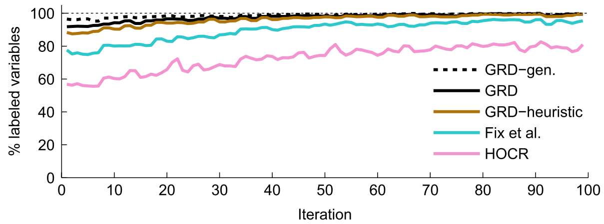
Ishikawa [15] used image denoising as a benchmark problem for higher-order pseudo-boolean minimization. In each iteration the proposals are generated in two possible ways which are alternated: by blurring the current image and picking all pixels at random. The smoothness term consists of a Fields of Experts (FoE) model using patches of size 2 × 2. Thus, quartic polynomials are needed to formulate the image restoration task as a pseudo-boolean minimization problem.

Figs. 2 and 3 show a comparison between the different methods for this problem. Generalized roof duality performed very well, often labeling very close to 100% of the problem variables. In the plots showing the number of persistencies in each iteration, we show the average over two types of proposals generated, just as in [8]. Otherwise, the oscillating graphs overlap and the plot becomes hard to read.

If we instead consider the energy as a function of time spent computing, our heuristic method still outperforms HOCR, but the difference is smaller. This is due to the fact that GRD has to solve multiple graph cut problems in each iteration while HOCR only has to solve one. The results are shown in Fig. 4. The best performing method is the one by Fix et al., which also solves just one graph cut problem in each iteration, but has better reductions than HOCR in general. In this application it does not pay off to iterate and compute the best possible solution; it is better to just generate a new proposal.



(a) Each method progresses independently.

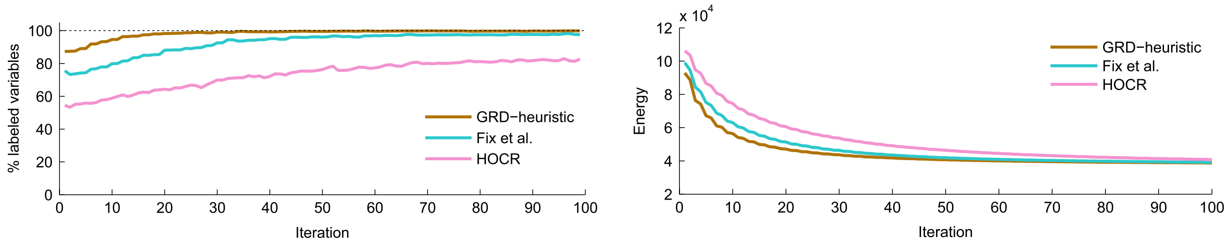


(b) Each method solves the same problem in each iteration.

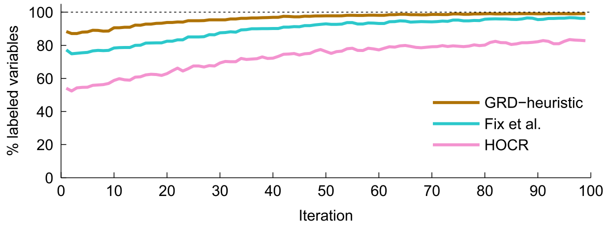
(c) Noisy image.

(d) Restored image.

Fig. 2. Restoring a small image. In each iteration a proposal is generated and each pixel can either stay the same or switch to the proposal. A quartic smoothness function is used.



(a) Each method progresses independently.



(b) Each method solves the same problem in each iteration.

(c) Noisy image.

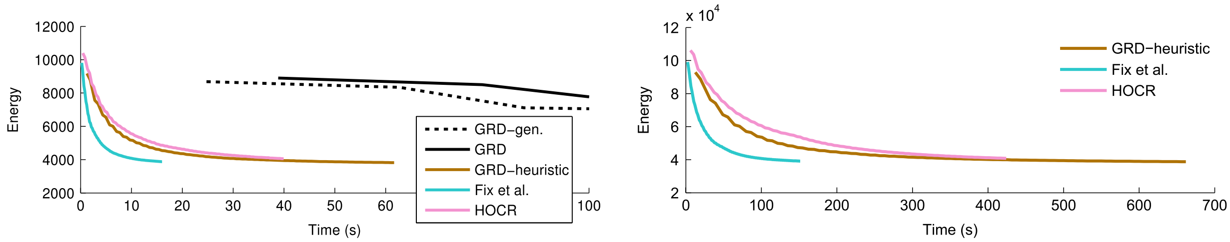
(d) Restored image.

Fig. 3. Restoring a larger image. Due to its size, we do not use LP-based relaxations.

8.2.2. *Stereo reconstruction*

In dense stereo reconstruction, second order surface priors have recently been used to obtain very good results [33]; see Fig. 5. The algorithm involves minimizing a number of pseudo-boolean functions of degree 3. Since the framework uses a heuristic to obtain a complete non-optimal solution (i.e. an upper bound), we instead compare to the HOCR reductions.

Table 2 shows the result for a few image sets. For this problem type, the simple heuristics does not perform better than the HOCR method.



(a) Problem in Fig. 2. (b) Problem in Fig. 3.

Fig. 4. Energy vs. time for the denoising experiments. The run times are the complete run times for all steps of the respective methods, i.e. linear programming, heuristics and minimum cut solvers are included. The method by Fix et al. [8] wins since it performs a single graph cut computation per iteration while having better reductions than HOCR.

(a) Image from input sequence. (b) Result. (c) Image from input sequence. (d) Result.

Fig. 5. From a calibrated sequence of images, stereo reconstruction can be used to recover a dense depth map [33].

Table 2
Comparison between HOCR and our methods for stereo reconstruction. The computed numbers are the sample means over all problem instances.

Set	Problems	n	Lower bounds $(\cdot 10^{10})$			Persistencies $(\cdot 10^{5})$		
			GRD	GRD-heuristic	HOCR	GRD	GRD-heuristic	HOCR
Cones	259	506,250	1.22974	1.22972	1.22945	4.998	4.917	4.986
Cloth3	392	462,870	0.833185	0.833176	0.833110	4.60	4.55	4.59

Table 3
Lower bounds on some 3-SAT problems from [14]. The instances are all satisfiable, so the optimal value is always 0. The computed numbers are the sample means $\pm$ one standard deviation.

Images	Problems	n	Lower bounds			Times (ms)		
			GRD	GRD-heur.	HOCR	GRD	GRD-heur.	HOCR
CBS	1000	100	-173 ± 2.90	-181 ± 2.63	-251 ± 4.25	77.54 ± 14.88	8.93 ± 10.34	2.20 ± 5.62
RTI	500	100	-183 ± 2.87	-192 ± 2.65	-268 ± 4.77	86.80 ± 16.73	8.26 ± 8.77	2.47 ± 5.60
uf20–91	1000	20	-26 ± 1.39	-31 ± 1.51	-54 ± 2.55	6.78 ± 8.15	1.21 ± 4.11	0.53 ± 2.83

8.3. 3-SAT

A clause "x_i or not x_j or not x_k" is encoded as the term $\bar{x}_i x_j x_k$. The energy function to be minimized is the sum of all these terms, for example,

$$f(\boldsymbol{x}) = \bar{x}_1 x_2 x_3 + x_1 \bar{x}_3 x_6 + \cdots + x_2 \bar{x}_3 \bar{x}_4. \tag{34}$$

If the problem is satisfiable, then the optimal value is 0. Note that by construction, 0 is always trivially a lower bound to the energy function. The optimization methods do not have this information, though. Hence, a bound lower than 0 will be reported in general.

Table 3 shows the performance on some publicly available SAT databases. Since persistencies are almost never found for these problems, we compare lower bounds only. Knowing how the problems are generated, it is trivial to prove a lower bound of 0. Though, this information about the problem structure is not available to any method in Table 3.

9. Concluding discussion

Generalized roof duality of arbitrary degree. We have seen three different approaches to construct submodular relaxations: (i) via recognizable sets $\Gamma_{\mathrm{sym},m}$, (ii) using generators, and (iii) heuristics. However, it is not straightforward to apply these approaches to higher-order ($m \geq 5$) cases. As the dimension goes up, things quickly become impractical due to the sheer size of the relaxation sets. So, restricting the computations to subsets are necessary. It is not clear how to achieve a good compromise between tightness of the lower bound and efficiency of the computations. Further, it is an open problem to characterize the symmetric submodular cone $\Gamma_{\text{sym},m} \subset \Gamma_{\text{suff},m}$ and to derive generators for the cone of expressible pseudo-boolean functions when $m \geq 5$. In [34], it is conjectured that the generators are given by the so-called *upper and lower fans*. These issues are left for future work. Of course, in practice one can work with any set of higher-order, expressible generators. The generators can be problem-specific and the number of them can depend on the computational resources available.

Another open problem is whether the cubic submodular relaxations can be improved by enlarging the set of relaxations to include bisubmodular and higher-order submodular relaxations. Theorem 6 tells us that when degree$(f) = 2$, it is enough to consider relaxations of the same degree. On the other hand, in [18], an example with degree$(f) = 4$ is given where a bisubmodular relaxation strictly dominates any submodular relaxation. We conjecture that when degree$(f) = 3$, the tightest cubic submodular relaxation dominates all other submodular relaxations of arbitrary degree.

Summary. We have shown how the roof duality bound for unconstrained quadratic pseudo-boolean functions can be generalized for higher-order functions. The bound is defined as the maximum lower bound over a set of submodular relaxations. Our main result is that a solution that attains this bound can be computed in polynomial time. By definition, the generalized roof dual bound is superior to many previously proposed reduction schemes.

The main focus of our analysis is on cubic and quartic submodular relaxations, which are most interesting from an application point of view. The cubic case is more straight-forward and the solution is more elegant than the quartic case, mainly due to the fact that all cubic submodular functions are expressible and the functions form a set which is recognizable.

The experimental results demonstrate that much better lower bounds, and many more labeled variables can be determined with the generalized roof dual bound compared to the state of the art. The price to pay is the computational effort due to the time spent on (i) constructing the relaxations, and (ii) the iterative improvements. The method is still very attractive in terms of speed, particularly for large-scale problems involving several thousands of variables. For $m = 4$, the roof dual based on generators is preferable to the approach using relaxations in $\Gamma_{\text{sym},4}$, both in terms of execution times and bounding performance.

Implementation. We have made our implementation publicly available at https://github.com/PetterS/submodular.

Acknowledgments

This work has been funded by the Swedish Foundation for Strategic Research (FFL and VINST) and by the European Research Council (GlobalVision grant no. 209480). We want to thank Carl Olsson for helpful discussions.

Appendix. Quartic submodularity

By first expanding all conjugate factors in the symmetric form (26) to a multilinear polynomial and then applying the sufficient condition (25), one obtains the following constraints.

Lemma 19. *A quartic symmetric polynomial g represented by* (26) *is submodular and expressible by a quadratic submodular polynomial if*

$$\begin{aligned}
b_{ij} \;&\leq (23\text{a}) - b^+_{ij\bullet\bullet} - b^+_{i\bullet j\bullet} - b^+_{i\bullet\bullet j} - b^+_{\bullet ij\bullet} - b^+_{\bullet i\bullet j} - b^+_{\bullet\bullet ij} - |c|_{ij\bullet\bullet} - |c|_{i\bullet j\bullet} - |c|_{\bullet ij\bullet} - |d|_{ij\bullet\bullet} - |d|_{i\bullet\bullet j} - |d|_{\bullet i\bullet j} \\
&\quad - |e|_{i\bullet j\bullet} - |e|_{i\bullet\bullet j} - |e|_{\bullet\bullet ij} - |p|_{\bullet ij\bullet} - |p|_{\bullet i\bullet j} - |p|_{\bullet\bullet ij} - q^+_{ij\bullet\bullet} - |q|_{ij\bullet\bullet} - q^+_{\bullet\bullet ij} \\
&\quad - r^+_{i\bullet j\bullet} - |r|_{i\bullet j\bullet} - r^+_{\bullet i\bullet j} - s^+_{i\bullet\bullet j} - |s|_{i\bullet\bullet j} - s^+_{\bullet ij\bullet} \\
-c_{ij} &\leq (23\text{b}) - c^-_{i\bullet\bullet j} - c^-_{\bullet i\bullet j} - c^-_{\bullet\bullet ij} - d^-_{i\bullet j\bullet} - d^-_{\bullet ij\bullet} - d^-_{\bullet\bullet ij} - e^-_{ij\bullet\bullet} - e^-_{\bullet ij\bullet} - e^-_{\bullet i\bullet j} - p^-_{ij\bullet\bullet} - p^-_{i\bullet j\bullet} - p^-_{i\bullet\bullet j} \\
&\quad - |q|_{i\bullet j\bullet} - |q|_{i\bullet\bullet j} - |q|_{\bullet ij\bullet} - |q|_{\bullet i\bullet j} - |r|_{ij\bullet\bullet} - |r|_{i\bullet\bullet j} - |r|_{\bullet ij\bullet} - |r|_{\bullet\bullet ij} - |s|_{ij\bullet\bullet} - |s|_{i\bullet j\bullet} - |s|_{\bullet i\bullet j} - |s|_{\bullet\bullet ij},
\end{aligned}$$

for $1 \leq i < j \leq n$*, where* (23a) *and* (23b) *denote the right-hand side of the inequalities, respectively.*

References

[1] A. Billionet, M. Minoux, Maximizing a supermodular pseudo-boolean function: a polynomial algorithm for cubic functions, Discrete Appl. Math. 12 (1985) 1–11.
[2] A. Billionnet, A. Sutter, Persistency in quadratic 0–1 optimization, Math. Program. 54 (1–3) (1992) 115–119.
[3] E. Boros, P.L. Hammer, Pseudo-boolean optimization, Discrete Appl. Math. 123 (2002) 155–225.
[4] E. Boros, P.L. Hammer, R. Sun, G. Tavares, A max-flow approach to improved lower bounds for quadratic unconstrained binary optimization (QUBO), Discrete Optim. 5 (2) (2008) 501–529.
[5] Y. Boykov, O. Veksler, R. Zabih, Fast approximate energy minimization via graph cuts, IEEE Trans. Pattern Anal. Mach. Intell. 23 (11) (2001) 1222–1239.
[6] Y. Crama, Recognition problems for special classes of polynomials in 0–1 variables, Math. Program. 44 (1–3) (1989) 139–155.
[7] D. Cremers, L. Grady, Statistical priors for efficient combinatorial optimization via graph cuts, in: European Conf. Computer Vision, Graz, Austria, 2006.
[8] A. Fix, A. Grubner, E. Boros, R. Zabih, A graph cut algorithm for higher-order Markov random fields, in: Int. Conf. Computer Vision, Barcelona, Spain, 2011.

[9] D. Freedman, P. Drineas, Energy minimization via graph cuts: settling what is possible, in: Conf. Computer Vision and Pattern Recognition, San Diego, USA, 2005.
[10] S. Fujishige, S. Iwata, Bisubmodular function minimization, SIAM J. Discrete Math. 19 (4) (2006) 1065–1073.
[11] A.C. Gallagher, D. Batra, D. Parikh, Inference for order reduction in Markov random fields, in: Conf. Computer Vision and Pattern Recognition, Colorado Springs, USA, 2011.
[12] F. Glover, B. Alidaee, C. Rego, G. Kochenberger, One-pass heuristics for large-scale unconstrained binary quadratic problems, European J. Oper. Res. 137 (2) (2002) 272–287.
[13] P.L. Hammer, P. Hansen, B. Simeone, Roof duality, complementation and persistency in quadratic 0–1 optimization, Math. Program. 28 (2) (1984) 121–155.
[14] H.H. Hoos, T. Stützle, SATLIB: an online resource for research on SAT, in: SAT 2000, IOS Press, 2000, pp. 283–292.
[15] H. Ishikawa, Transformation of general binary MRF minimization to the first order case, IEEE Trans. Pattern Anal. Mach. Intell. 33 (6) (2011) 1234–1249.
[16] F. Kahl, P. Strandmark, Generalized roof duality for pseudo-boolean functions, in: Int. Conf. Computer Vision, Barcelona, Spain, 2011.
[17] P. Kohli, M.P. Kumar, P.H.S. Torr, P^3 & beyond: move making algorithms for solving higher order functions, IEEE Trans. Pattern Anal. Mach. Intell. 31 (9) (2009) 1645–1656.
[18] V. Kolmogorov, Generalized roof duality and bisubmodular functions, Discrete Appl. Math. 160 (4–5) (2012) 416–426.
[19] V. Kolmogorov, C. Rother, Minimizing nonsubmodular functions with graph cuts—a review, IEEE Trans. Pattern Anal. Mach. Intell. 29 (7) (2007) 1274–1279.
[20] N. Komodakis, N. Paragios, Beyond pairwise energies: efficient optimization for higher-order MRFs, in: Conf. Computer Vision and Pattern Recognition, Miami, USA, 2009.
[21] L. Ladicky, C. Russell, P. Kohli, P.H.S. Torr, Graph cut based inference with co-occurrence statistics, in: European Conf. Computer Vision, Crete, Greece, 2010.
[22] X. Lan, S. Roth, D.P. Huttenlocher, M.J. Black, Efficient belief propagation with learned higher-order Markov random fields, in: European Conf. Computer Vision, Graz, Austria, 2006.
[23] V. Lempitsky, C. Rother, S. Roth, A. Blake, Fusion moves for Markov random field optimization, IEEE Trans. Pattern Anal. Mach. Intell. 32 (8) (2010) 1392–1405.
[24] L. Lovász, Submodular functions and convexity, Mathematical Programming: The State of the Art (1983) 235–257.
[25] S.H. Lu, A.C. Williams, Roof duality for polynomial 0–1 optimization, Math. Program. 37 (3) (1987) 357–360.
[26] S. Nowozin, C.H. Lampert, Global interactions in random field models: a potential function ensuring connectedness, SIAM J. Imaging Sciences 3 (4) (2010) 1048–1074.
[27] S.D. Promislow, V.R. Young, Supermodular functions on finite lattices, Order 22 (4) (2005) 389–413.
[28] C. Rother, P. Kohli, W. Feng, J. Jia, Minimizing sparse higher order energy functions of discrete variables, in: Conf. Computer Vision and Pattern Recognition, Miami, USA, 2009.
[29] T. Schoenemann, F. Kahl, D. Cremers, Curvature regularity for region-based image segmentation and inpainting: a linear programming relaxation, in: Int. Conf. Computer Vision, Kyoto, Japan, 2009.
[30] P. Strandmark, F. Kahl, Curvature regularization for curves and surfaces in a global optimization framework, in: Energy Minimization Methods in Computer Vision and Pattern Recognition, in: Lecture Notes in Computer Science, vol. 6819, Springer, 2011.
[31] P. Strandmark, F. Kahl, Parallel and distributed graph cuts by dual decomposition, in: Conf. Computer Vision and Pattern Recognition, San Francisco, USA, 2010.
[32] T. Werner, High-arity interactions, polyhedral relaxations, and cutting plane algorithm for MAP-MRF, in: Conf. Computer Vision and Pattern Recognition, Anchorage, USA, 2008.
[33] O.J. Woodford, P.H.S. Torr, I.D. Reid, A.W. Fitzgibbon, Global stereo reconstruction under second-order smoothness priors, IEEE Trans. Pattern Anal. Mach. Intell. 31 (12) (2009) 2115–2128.
[34] S. Živný, D.A. Cohen, P.G. Jeavons, The expressive power of binary submodular functions, Discrete Appl. Math. 157 (15) (2009) 3347–3358.
[35] S. Živný, P.G. Jeavons, Classes of submodular constraints expressible by graph cuts, Constraints 15 (3) (2010) 430–452.
[36] S. Živný, P.G. Jeavons, Which submodular functions are expressible using binary submodular functions? Technical Report CS-RR-08-08, Oxford University Computing Laboratory, 2008.