The Expressive Power of Binary Submodular Functions*

Stanislav Živný[1], David A. Cohen[2], and Peter G. Jeavons[1]

[1] Computing Laboratory, University of Oxford, UK
{stanislav.zivny,peter.jeavons}@comlab.ox.ac.uk
[2] Department of Computer Science, Royal Holloway, University of London, UK
d.cohen@rhul.ac.uk

Abstract. We investigate whether all Boolean submodular functions can be decomposed into a sum of binary submodular functions over a possibly larger set of variables. This question has been considered within several different contexts in computer science, including computer vision, artificial intelligence, and pseudo-Boolean optimisation. Using a connection between the expressive power of valued constraints and certain algebraic properties of functions, we answer this question negatively.
Our results have several corollaries. First, we characterise precisely which submodular polynomials of arity 4 can be expressed by binary submodular polynomials. Next, we identify a novel class of submodular functions of arbitrary arities which can be expressed by binary submodular functions, and therefore minimised efficiently using a so-called expressibility reduction to the Min-Cut problem. More importantly, our results imply limitations on this kind of reduction and establish for the first time that it cannot be used in general to minimise arbitrary submodular functions. Finally, we refute a conjecture of Promislow and Young on the structure of the extreme rays of the cone of Boolean submodular functions.
Keywords: Decomposition of submodular functions, Min-Cut, Pseudo-Boolean optimisation, Submodular function minimisation.

1 Introduction

A function $f : 2^V \to \mathbb{R}$ is called *submodular* if for all $S, T \subseteq V$,

$$f(S \cap T) + f(S \cup T) \ \leq \ f(S) + f(T).$$

Submodular functions are a key concept in operational research and combinatorial optimisation [27,26,33,32,14,21,17]. Examples include cut capacity functions, matroid rank functions, and entropy functions. Submodular functions are often considered to be a discrete analogue of convex functions [24].

Both minimising and maximising submodular functions, possibly under some additional conditions, have been considered extensively in the literature. Submodular function maximisation is easily shown to be NP-hard [32] since it generalises many standard NP-hard problems such as the maximum cut problem. In contrast, the problem of *minimising* a submodular function (SFM) can be solved efficiently with only polynomially many oracle calls, see [17]. The time complexity of the fastest known general algorithm for SFM is $O(n^6 + n^5 L)$,

* The authors would like to thank Martin Cooper for fruitful discussions on submodular functions and in particular for help with the proof of Theorem 1. The first author gratefully acknowledges the support of EPSRC grant EP/F01161X/1.

where n is the number of variables and L is the time required to evaluate the function [28].

The minimisation of submodular functions on sets is equivalent to the minimisation of submodular functions on distributive lattices [32]. Krokhin and Larose have also studied the more general problem of minimising submodular functions on non-distributive lattices [22].

An important and well-studied sub-problem of SFM is the minimisation of submodular functions of bounded arity (SFM_b), also known as *locally defined* submodular functions [8], or submodular functions with *succinct representation* [12]. In this scenario the submodular function to be minimised is defined as the sum of a collection of functions which each depend only on a bounded number of variables. Locally defined optimisation problems of this kind occur in a wide variety of contexts:

- In the context of pseudo-Boolean optimisation, such problems involve the minimisation of Boolean polynomials of bounded *degree* [2].
- In the context of artificial intelligence, they have been studied as *valued constraint satisfaction problems* (VCSP) [31], also known as *soft* or *weighted* constraint satisfaction problems.
- In the context of computer vision, such problems are often formulated as *Gibbs energy minimisation* problems or *Markov Random Fields* (also known as *Conditional Random Fields*) [23].

We will present our results primarily in the language of pseudo-Boolean optimisation. Hence an instance of SFM_b with n variables will be represented as a polynomial in n Boolean variables, of some fixed bounded degree.

However, the concept of submodularity is important in a wide variety of fields within computer science, and our results have direct consequences for Constraint Satisfaction Problems [10,7,19,11] and Computer Vision [20]. Due to space restrictions we will not elaborate on these connections.

A general algorithm for SFM can always be used for the more restricted SFM_b, but the special features of this more restricted problem sometimes allow more efficient special-purpose algorithms to be used. (Note that we are focusing on *exact* algorithms which find an optimal solution.) In particular, it has been shown that certain cases can be solved much more efficiently by reducing to the Min-Cut problem; that is, the problem of finding a minimum cut in a directed graph which includes a given source vertex and excludes a given target vertex. For example, it has been known since 1965 that the minimisation of *quadratic* submodular polynomials is equivalent to finding a minimum cut in a corresponding directed graph [16,2]. Hence quadratic submodular polynomials can be minimised in $O(n^3)$ time, where n is the number of variables.

A Boolean polynomial in at most 2 variables has degree at most 2, so any *sum* of binary Boolean polynomials has degree at most 2; in other words, it is quadratic. It follows that an efficient algorithm, based on reduction to Min-Cut, can be used to minimise any class of functions that can be written as a sum of binary submodular polynomials. We will say that a polynomial that can be written in this way, perhaps with additional variables to be minimised over, is *expressible* by binary submodular polynomials (see Section 2.1). The following classes of functions have all been shown to be expressible by binary submodular polynomials in this way[3], over the past four decades:

[3] In fact, it is known that *all* Boolean polynomials (of arbitrary degree) are expressible by binary polynomials [2], but the general construction does not preserve submodularity; that is, the resulting binary polynomials are not necessarily submodular.

– polynomials where all terms of degree 2 or more have negative coefficients (also known as *negative-positive* polynomials) [30];
– cubic submodular polynomials [1];
– $\{0,1\}$-valued submodular functions (also known as 2-monotone functions) [10,6];
– a class recently found by Živný and Jeavons [35] and independently in [34].

All these classes of functions have been shown to be expressible by binary submodular polynomials and hence minimisable in cubic time (in the total number of variables). Moreover, several broad classes of submodular functions over non-Boolean domains have also been shown to be expressible by binary submodular functions and hence minimisable in cubic time [3,5,6]. This series of positive expressibility results naturally raises the following question:

Question 1. Are *all* submodular polynomials expressible by binary submodular polynomials, over a possibly larger set of variables?

Each of the above expressibility results was obtained by an ad-hoc construction, and no general technique[4] has previously been proposed which is sufficiently powerful to address Question 1.

1.1 Contributions

Cohen et al. recently developed a novel algebraic approach to characterising the expressive power of valued constraints in terms of certain algebraic properties of those constraints [4].

Using this systematic algebraic approach we are able to give a negative answer to Question 1: we show that there exist submodular polynomials of degree 4 that cannot be expressed by binary submodular polynomials. More precisely, we characterise exactly which submodular polynomials of arity 4 are expressible by binary submodular polynomials and which are not.

On the way to establishing these results we show that two broad families of submodular functions, known as *upper fans* and *lower fans*, are all expressible by binary submodular functions. This provides a new class of submodular polynomials of all arities which are expressible by binary submodular polynomials and hence solvable efficiently by reduction to Min-Cut. We use the expressibility of this family, and the existence of non-expressible functions, to refute a conjecture from [29] on the structure of the extreme rays of the cone of Boolean submodular functions, and suggest a more refined conjecture of our own.

2 Preliminaries

In this section, we introduce the basic definitions and the main tools used throughout the paper.

2.1 Cost functions and expressibility

We denote by $\overline{\mathbb{R}}$ the set of all real numbers together with (positive) infinity. For any fixed set D, a function ϕ from D^n to $\overline{\mathbb{R}}$ will be called a *cost function* on D of arity n. If the range of ϕ lies entirely within $\mathbb{R}$, then ϕ is called a *finite-valued*

[4] For example, standard combinatorial counting techniques cannot resolve this question because we allow arbitrary real-valued coefficients in submodular polynomials. We also allow an arbitrary number of additional variables.

cost function. If the range of ϕ is $\{0, \infty\}$, then ϕ can be viewed as a predicate, or *relation*, allowing just those tuples $t \in D^n$ for which $\phi(t) = 0$.

Cost functions can be added and multiplied by arbitrary real values, hence for any given set of cost functions, Γ, we define the convex cone generated by Γ, as follows.

Definition 1. *For any set of cost functions Γ, the* cone generated by Γ*, denoted* $\mathrm{Cone}(\Gamma)$*, is defined by:*

$$\mathrm{Cone}(\Gamma) = \{\alpha_1\phi_1 + \cdots + \alpha_r\phi_r \mid r \geq 1;\ \phi_1, \ldots, \phi_r \in \Gamma;\ \alpha_1, \ldots, \alpha_r \geq 0\}.$$

Definition 2. *A cost function ϕ of arity n is said to be* expressible *by a set of cost functions Γ if $\phi = \min_{y_1,\ldots,y_j} \phi'(x_1, \ldots, x_n, y_1, \ldots, y_j) + \kappa$, for some $\phi' \in \mathrm{Cone}(\Gamma)$ and some constant κ.*

The variables $y_1, \ldots, y_j$ are called extra *(or* hidden*) variables, and ϕ' is called a* gadget *for ϕ over Γ.*

We denote by $\langle\Gamma\rangle$ the *expressive power* of Γ, which is the set of all cost functions expressible by Γ.

It was shown in [4] that the expressive power of a set of cost functions is characterised by certain algebraic properties of those cost functions called fractional polymorphisms. For the results of this paper, we will only need a certain subset of these algebraic properties, called *multimorphisms* [7]. These are defined in Definition 3 below (see also Figure 1).

The i-th component of a tuple t will be denoted by $t[i]$. Note that any operation on a set D can be extended to tuples over the set D in a standard way, as follows. For any function $f : D^k \to D$, and any collection of tuples $t_1, \ldots, t_k \in D^n$, define $f(t_1, \ldots, t_k) \in D^n$ to be the tuple $\langle f(t_1[1], \ldots, t_k[1]), \ldots, f(t_1[n], \ldots, t_k[n])\rangle$.

Definition 3 ([7]). *Let $\mathcal{F} : D^k \to D^k$ be the function whose k-tuple of output values is given by the tuple of functions $\mathcal{F} = \langle f_1, \ldots, f_k\rangle$, where each $f_i : D^k \to D$.*

For any n-ary cost function ϕ, we say that $\mathcal{F}$ is a k-ary multimorphism *of ϕ if, for all $t_1, \ldots, t_k \in D^n$,*

$$\sum_{i=1}^{k} \phi(t_i) \ \geq\ \sum_{i=1}^{k} \phi(f_i(t_1, \ldots, t_k)).$$

For any set of cost functions, Γ, we will say that $\mathcal{F}$ is a multimorphism of Γ if $\mathcal{F}$ is a multimorphism of every cost function in Γ. The set of all multimorphisms of Γ will be denoted $\mathsf{Mul}(\Gamma)$.

Note that multimorphisms are preserved under expressibility. In other words, if $\mathcal{F} \in \mathsf{Mul}(\Gamma)$, and $\phi \in \langle\Gamma\rangle$, then $\mathcal{F} \in \mathsf{Mul}(\{\phi\})$ [7,4]. This has two important corollaries. First, if $\langle\Gamma_1\rangle = \langle\Gamma_2\rangle$, then $\mathsf{Mul}(\Gamma_1) = \mathsf{Mul}(\Gamma_2)$. Second, if there exists $\mathcal{F} \in \mathsf{Mul}(\Gamma)$ such that $\mathcal{F} \notin \mathsf{Mul}(\{\phi\})$, then ϕ is not expressible by Γ, that is, $\phi \notin \langle\Gamma\rangle$.

2.2 Lattices and submodularity

Recall that L is a *lattice* if L is a partially ordered set in which every pair of elements (a, b) has a unique supremum and a unique infimum. For a finite lattice L and a pair of elements (a, b), we will denote the unique supremum of a and b by $a \vee b$, and the unique infimum of a and b by $a \wedge b$.

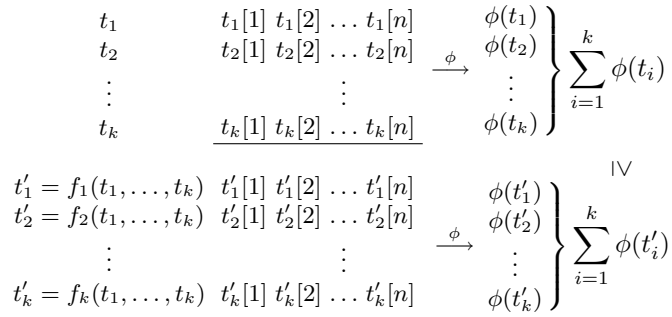

Fig. 1. Inequality establishing $\mathcal{F} = \langle f_1, \ldots, f_k \rangle$ as a multimorphism of cost function ϕ (see Definition 3).

For any finite lattice-ordered set D, a cost function $\phi : D^n \to \overline{\mathbb{R}}$ is called *submodular* if for every $u, v \in D^n$, $\phi(u \wedge v) + \phi(u \vee v) \leq \phi(u) + \phi(v)$ where both $\wedge$ and $\vee$ are applied coordinate-wise on tuples u and v [27]. This standard definition can be reformulated very simply in terms of multimorphisms: ϕ is submodular if $\langle \wedge, \vee \rangle \in \mathsf{Mul}(\{\phi\})$.

Using results from [7] and [32], it can be shown that any submodular cost function ϕ can be expressed as the sum of a finite-valued submodular cost function ϕ_{fin}, and a submodular relation ϕ_{rel}, that is, $\phi = \phi_{\mathsf{fin}} + \phi_{\mathsf{rel}}$.

Moreover, it is known that all submodular *relations* are binary decomposable (that is, equal to the sum of their binary projections) [18], and hence expressible by binary submodular relations. Therefore, when considering which cost functions are expressible by binary submodular cost functions, we can restrict our attention to *finite-valued* cost functions without any loss of generality.

Next we define some particular families of submodular cost functions, first described in [29], which will turn out to play a central role in our analysis.

Definition 4. *Let L be a lattice. We define the following cost functions on L:*

- *For any set A of pairwise incomparable elements $\{a_1, \ldots, a_m\} \subseteq L$, such that each pair of distinct elements (a_i, a_j) has the same least upper bound, $\bigvee A$, the following cost function is called an* upper fan*:*

$$\phi_A(x) = \begin{cases} -2 & \text{if } x \geq \bigvee A, \\ -1 & \text{if } x \not\geq \bigvee A, \text{ but } x \geq a_i \text{ for some } i, \\ 0 & \text{otherwise.} \end{cases}$$

- *For any set B of pairwise incomparable elements $\{a_1, \ldots, a_m\} \subseteq L$, such that each pair of distinct elements (a_i, a_j) has the same greatest lower bound, $\bigwedge B$, the following cost function is called a* lower fan*:*

$$\phi_B(x) = \begin{cases} -2 & \text{if } x \leq \bigwedge B, \\ -1 & \text{if } x \not\leq \bigwedge B, \text{ but } x \leq a_i \text{ for some } i, \\ 0 & \text{otherwise.} \end{cases}$$

We call a cost function a *fan* if it is either an upper fan or a lower fan. Note that our definition of fans is slightly more general than the definition in [29]. In particular, we allow the set A to be empty, in which case the corresponding upper fan ϕ_A is a constant function. It is not hard to show that all fans are submodular [29].

2.3 Boolean cost functions and polynomials

In this paper we will focus on problems over Boolean domains, that is, where $D = \{0, 1\}$.

Any cost function of arity n can be represented as a table of values of size D^n. Moreover, a finite-valued cost function $\phi : D^n \to \mathbb{R}$ on a Boolean domain $D = \{0, 1\}$ can also be represented as a unique *polynomial* in n (Boolean) variables with coefficients from $\mathbb{R}$ (such functions are sometimes called *pseudo-Boolean functions* [2]). Hence, in what follows, we will often refer to a finite-valued cost function on a Boolean domain and its corresponding polynomial interchangeably.

For polynomials over Boolean variables there is a standard way to define *derivatives* of each order (see [2]). For example, the second-order derivative of a polynomial p, with respect to the first two indices, denoted $\delta_{1,2}(\mathbf{x})$, is defined as $p(1, 1, \mathbf{x}) - p(1, 0, \mathbf{x}) - p(0, 1, \mathbf{x}) + p(0, 0, \mathbf{x})$. Derivatives for other pairs of indices are defined analogously. It was shown in [13] that a polynomial $p(x_1, \ldots, x_n)$ over Boolean variables $x_1, \ldots, x_n$ represents a submodular cost function if, and only if, its second-order derivatives $\delta_{i,j}(\mathbf{x})$ are non-positive for all $1 \leq i < j \leq n$ and all $\mathbf{x} \in D^{n-2}$. An immediate corollary is that a quadratic polynomial represents a submodular cost function if, and only if, the coefficients of all quadratic terms are non-positive.

Note that a cost function is called *supermodular* if all its second-order derivatives are non-negative. Clearly, f is submodular if, and only if, $-f$ is supermodular, so it is straightforward to translate results about supermodular functions, such as those given in [6] and [29], into similar results for submodular functions, and we will use this observation several times below. Cost functions which are both submodular and supermodular (in other words, all second-order derivatives are equal to zero) are called *modular*, and polynomials corresponding to modular cost functions are linear [2].

Example 1. For any set of indices $I = \{i_1, \ldots, i_m\} \subseteq \{1, \ldots, n\}$ we can define a cost function ϕ_I in n variables as follows:

$$\phi_I(x_1, \ldots, x_n) \;=\; \begin{cases} -1 & \text{if } (\forall i \in I)(x_i = 1), \\ 0 & \text{otherwise.} \end{cases}$$

The polynomial representation of ϕ_I is $p(x_1, \ldots, x_n) = -x_{i_1} \ldots x_{i_m}$, which is a polynomial of degree m. Note that it is straightforward to verify that ϕ_I is submodular by checking the second-order derivatives of p.

However, the function ϕ_I is also expressible by *binary* submodular polynomials, using a single extra variable, y, as follows:

$$\phi_I(x_1, \ldots, x_n) = \min_{y \in \{0,1\}} \{-y + y \sum_{i \in I} (1 - x_i)\}.$$

We remark that this is a special case of the expressibility result for negative-positive polynomials first obtained in [30].

Note that when $D = \{0, 1\}$, the set D^n with the product ordering is isomorphic to the lattice of all subsets of an n-element set ordered by inclusion. Hence, a cost function on a Boolean domain can be viewed as a cost function defined on a lattice of subsets, and we can apply Definition 4 to identify certain Boolean functions as upper fans or lower fans, as the following example indicates.

Example 2. Let $A = \{I_1, \ldots, I_r\}$ be a set of subsets of $\{1, 2, \ldots, n\}$ such that for all $i \neq j$ we have $I_i \not\subseteq I_j$ and $I_i \cup I_j = \bigcup A$.

By Definition 4, the corresponding upper fan function ϕ_A has the following polynomial representation:

$$p(x_1, \ldots, x_n) = (r-2) \prod_{i \in \bigcup A} x_i - \prod_{i \in I_1} x_i - \cdots - \prod_{i \in I_r} x_i.$$

We remark that any permutation of a set D gives rise to an automorphism of cost functions over D. In particular, for any cost function f on a Boolean domain D, the *dual* of f is the corresponding cost function which results from exchanging the values 0 and 1 for all variables. In other words, if p is the polynomial representation of f, then the dual of f is the cost function whose polynomial representation is obtained from p by replacing all variables x with $1-x$. Observe that, due to symmetry, taking the dual preserves submodularity and expressibility by binary submodular cost functions.

It is not hard to see that upper fans are duals of lower fans and vice versa.

3 Results

In this section, we present our main results. First, we show that fans of all arities are expressible by binary submodular cost functions. Next, we characterise the multimorphisms of binary submodular cost functions. Combining these results, we then characterise precisely which 4-ary submodular cost functions are expressible by binary submodular cost functions. More importantly, we show that some submodular cost functions are *not* expressible by binary submodular cost functions, and therefore cannot be minimised using the Min-Cut problem via an expressibility reduction. Finally, we consider the complexity of recognizing which cost functions are expressible by binary submodular cost functions.

3.1 Expressibility of upper fans and lower fans

We denote by $\Gamma_{\mathsf{sub},n}$ the set of all finite-valued submodular cost functions of arity at most n on a Boolean domain D, and we set $\Gamma_{\mathsf{sub}} = \bigcup_n \Gamma_{\mathsf{sub},n}$.

We denote by $\Gamma_{\mathsf{fans},n}$ the set of all fans of arity at most n on a Boolean domain D, and we set $\Gamma_{\mathsf{fans}} = \bigcup_n \Gamma_{\mathsf{fans},n}$.

Our next result shows that $\Gamma_{\mathsf{fans}} \subseteq \langle \Gamma_{\mathsf{sub},2} \rangle$. The proof is omitted due to space restrictions.

Theorem 1. *Any fan on a Boolean domain D is expressible by binary submodular functions on D using at most $1 + \lfloor m/2 \rfloor$ extra variables, where m is the degree of its polynomial representation.*

Many of the earlier expressibility results mentioned in Section 1 can be obtained as simple corollaries of Theorem 1, as the following examples indicate.

Example 3. Any negative monomial $-x_1 x_2 \cdots x_m$ is a positive multiple of an upper fan, and the positive linear monomial x_1 is equal to $-(1 - x_1) + 1$, so it is a positive multiple of a lower fan, plus a constant. Hence all negative-positive submodular polynomials are contained in $\mathrm{Cone}(\Gamma_{\mathsf{fans}})$, and by Theorem 1, they are expressible by binary submodular polynomials, as originally shown in [30].

Example 4. A polynomial is called *homogeneous* [1] or *polar* [9] if it can be expressed as a sum of terms of the form $ax_1x_2\dots x_k$ or $a(1-x_1)(1-x_2)\dots(1-x_k)$ with positive coefficients a, together with a constant term. It was observed in [1] that all polar polynomials are supermodular, so all negated polar polynomials are submodular. As every negated term $-ax_1x_2\dots x_k$, is a positive multiple of an upper fan, and every negated term $-a(1-x_1)(1-x_2)\dots(1-x_k)$, is a positive multiple of a lower fan, by Theorem 1, all cost functions which are the negations of polar polynomials are expressible by binary submodular polynomials, and hence solvable by reduction to Min-Cut, as originally shown in [1].

Example 5. Any cubic submodular polynomial can be expressed as a positive sum of upper fans [29]. Hence, by Theorem 1, all cubic submodular polynomials are expressible by binary submodular polynomials, as originally shown in [1].

Example 6. A Boolean cost function ϕ is called *2-monotone* [10] if there exist two sets $R, S \subseteq \{1,\dots,n\}$ such that $\phi(\mathbf{x}) = 0$ if $R \subseteq \mathbf{x}$ or $\mathbf{x} \subseteq S$ and $\phi(\mathbf{x}) = 1$ otherwise (where $R \subseteq \mathbf{x}$ means $\forall i \in R, x[i] = 1$ and $\mathbf{x} \subseteq S$ means $\forall i \notin S, x[i] = 0$). It was shown in [6, Proposition 2.9] that a 2-valued Boolean cost function is 2-monotone if, and only if, it is submodular.

For any 2-monotone cost function defined by the sets of indices R and S, it is straightforward to check that $\phi = \min_{y\in\{0,1\}} y(1+\phi_A/2) + (1-y)(1+\phi_B/2)$ where ϕ_A is the upper fan defined by $A = \{R\}$ and ϕ_B is the lower fan defined by $B = \{\overline{S}\}$. Note that the function $y\phi_A$ is an upper fan, and the function $(1-y)\phi_B$ is a lower fan. Hence, by Theorem 1, all 2-monotone polynomials are expressible by binary submodular polynomials, and solvable by reduction to Min-Cut, as originally shown in [10].

However, Theorem 1 also provides many new functions of all arities which have not previously been shown to be expressible by binary submodular functions, as the following example indicates.

Example 7. The function $2x_1x_2x_3x_4 - x_1x_2x_3 - x_1x_2x_4 - x_1x_3x_4 - x_2x_3x_4$ belongs to $\Gamma_{\mathsf{fans},4}$, but does not belong to any class of submodular functions which has previously been shown to be expressible by binary submodular functions. In particular, it does not belong to the class Γ_{new} identified in [34,35].

3.2 Characterisation of $\mathsf{Mul}(\Gamma_{\mathsf{sub},2})$

Since we have seen that a cost function can only be expressed by a given set of cost functions if it has the same multimorphisms, we now investigate the multimorphisms of $\Gamma_{\mathsf{sub},2}$.

A function $\mathcal{F} : D^k \to D^k$ is called *conservative* if, for each possible choice of $x_1,\dots,x_k$, the tuple $\mathcal{F}(x_1,\dots,x_k)$ is a permutation of $x_1,\dots,x_k$ (though different inputs may be permuted in different ways).

For any two tuples $\mathbf{x} = \langle x_1,\dots,x_k\rangle$ and $\mathbf{y} = \langle y_1,\dots,y_k\rangle$ over D, we denote by $H(\mathbf{x},\mathbf{y})$ the *Hamming distance* between $\mathbf{x}$ and $\mathbf{y}$, which is the number of positions at which the corresponding values are different.

Theorem 2. *For any Boolean domain D, and any $\mathcal{F} : D^k \to D^k$, the following are equivalent:*

1. $\mathcal{F} \in \mathsf{Mul}(\Gamma_{\mathsf{sub},2})$.
2. $\mathcal{F} \in \mathsf{Mul}(\Gamma^{\infty}_{\mathsf{sub},2})$, *where $\Gamma^{\infty}_{\mathsf{sub},2}$ denotes the set of binary submodular cost functions taking finite or infinite values.*
3. $\mathcal{F}$ *is conservative and Hamming distance non-increasing.*

The proof is omitted due to space restrictions.

3.3 Non-expressibility of Γ_{sub} over $\Gamma_{\mathsf{sub},2}$

Theorem 2 characterises the multimorphisms of $\Gamma_{\mathsf{sub},2}$, and hence enables us to systematically search (for example, using MATHEMATICA) for multimorphisms of $\Gamma_{\mathsf{sub},2}$ which are not multimorphisms of Γ_{sub}. In this way, we have identified the function $\mathcal{F}_{sep} : \{0,1\}^5 \to \{0,1\}^5$ defined in Figure 2. We will show in this section that this function can be used to characterise all the submodular functions of arity 4 which are expressible by binary submodular functions on a Boolean domain. Using this result, we show that some submodular functions are *not* expressible in this way.

$\mathbf{x}$	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 0 0 0 0 0 1 1 1 1 1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 0 0 0 0 1 1 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 0 1 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1 0 1
$\mathcal{F}_{sep}(\mathbf{x})$	0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 1 1 0 0 0 1 0 0 1 1 0 0 0 0 0 1 1 1 1 1 1 1 1 1 1 1 0 0 0 1 0 1 0 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1 0 1 1 1 1 1 1 1

Fig. 2. Definition of $\mathcal{F}_{sep}$.

Proposition 1. *$\mathcal{F}_{sep}$ is conservative and Hamming distance non-increasing.*

Proof. Straightforward exhaustive verification. □

Theorem 3. *For any function $f \in \Gamma_{\mathsf{sub},4}$ the following are equivalent:*

1. $f \in \langle \Gamma_{\mathsf{sub},2} \rangle$.
2. $\mathcal{F}_{sep} \in \mathsf{Mul}(\{f\})$.
3. $f \in \mathrm{Cone}(\Gamma_{\mathsf{fans},4})$.

Proof. First, we show (1) ⇒ (2). Proposition 1 and Theorem 2 imply that $\mathcal{F}_{sep}$ is a multimorphism of any binary submodular function on a Boolean domain. Hence having $\mathcal{F}_{sep}$ as a multimorphism is a necessary condition for any submodular cost function on a Boolean domain to be expressible by binary submodular cost functions.

Next, we show (2) ⇒ (3). Consider the complete set of inequalities on the values of a 4-ary cost function resulting from having the multimorphism $\mathcal{F}_{sep}$, as specified in Definition 3. A routine calculation in MATHEMATICA shows that, out of 16^5 such inequalities, there are 4635 which are distinct. After removing from these all those which are equal to the sum of two others, we obtain a system of just 30 inequalities which must be satisfied by any 4-ary submodular cost function which has the multimorphism $\mathcal{F}_{sep}$. Using the double description method [25], we obtain from these 30 inequalities an equivalent set of 31 extreme rays which generate the same polyhedral cone of cost functions. These extreme rays all correspond to fans or sums of fans.

Finally, we show (3) ⇒ (1). By Theorem 1, all fans are expressible over $\Gamma_{\mathsf{sub},2}$. It follows that any cost function in this cone of functions is also expressible over $\Gamma_{\mathsf{sub},2}$. □

Next we show that there are indeed 4-ary submodular cost functions which do not have $\mathcal{F}_{sep}$ as a multimorphism and therefore are not expressible by binary submodular cost functions.

Definition 5. *For any Boolean tuple t of arity 4 containing exactly 2 ones and 2 zeros, we define the 4-ary cost function θ_t as follows:*

$$\theta_t(x_1,x_2,x_3,x_4) \;=\; \begin{cases} -1 & \text{if } (x_1,x_2,x_3,x_4) = (1,1,1,1) \text{ or } (0,0,0,0), \\ 1 & \text{if } (x_1,x_2,x_3,x_4) = t, \\ 0 & \text{otherwise.} \end{cases}$$

Cost functions of the form θ_t were introduced in [29], where they are called *quasi-indecomposable* functions. We denote by Γ_{qin} the set of all (six) quasi-indecomposable cost functions of arity 4. It is straightforward to check that they are submodular, but the next result shows that they are *not* expressible by binary submodular functions.

Proposition 2. *For all $\theta \in \Gamma_{\mathsf{qin}}$, $\mathcal{F}_{sep} \notin \mathsf{Mul}(\{\theta\})$.*

Proof. The following table shows that $\mathcal{F}_{sep} \notin \mathsf{Mul}(\{\theta_{(1,1,0,0)}\})$.

$$\mathcal{F}_{sep}\left\{\begin{array}{cc} 1\,0\,1\,0 & 0 \\ 1\,0\,0\,1 & 0 \\ 0\,1\,0\,1 & 0 \\ 0\,1\,1\,0 & 0 \\ 0\,0\,1\,1 & 0 \\ \hline 0\,0\,1\,0 & 0 \\ 0\,0\,0\,1 & 0 \\ 1\,1\,0\,0 & 1 \\ 1\,0\,1\,1 & 0 \\ 0\,1\,1\,1 & 0 \end{array}\right. \begin{array}{c} \xrightarrow{\theta_{(1,1,0,0)}} \\ \\ \\ \\ \\ \\ \xrightarrow{\theta_{(1,1,0,0)}} \\ \\ \\ \end{array} \left.\begin{array}{l} \\ \\ \\ \\ \\ \sum = 0 \\ \\ \\ \\ \\ \sum = 1 \end{array}\right.$$

Permuting the columns appropriately establishes the result for all other $\theta \in \Gamma_{\mathsf{qin}}$. □

Corollary 1. *For all $\theta \in \Gamma_{\mathsf{qin}}$, $\theta \notin \langle \Gamma_{\mathsf{sub},2} \rangle$.*

Proof. By Theorem 3 and Proposition 2. □

Are there any other 4-ary submodular cost functions which are not expressible over $\Gamma_{\mathsf{sub},2}$? Promislow and Young characterised the extreme rays of the cone of all 4-ary submodular cost functions and established that $\Gamma_{\mathsf{sub},4} = \mathrm{Cone}(\Gamma_{\mathsf{fans},4} \cup \Gamma_{\mathsf{qin}})$ – see Theorem 5.2 of [29]. Hence the results in this section characterise the expressibility of all 4-ary submodular functions.

Promislow and Young conjectured that for $k \neq 4$, all extreme rays of $\Gamma_{\mathsf{sub},k}$ are fans [29]; that is, they conjectured that for all $k \neq 4$, $\Gamma_{\mathsf{sub},k} = \mathrm{Cone}(\Gamma_{\mathsf{fans},k})$. However, if this conjecture were true it would imply that all submodular functions of arity 5 and above were expressible by binary submodular functions, by Theorem 1. This is clearly not the case, because inexpressible cost functions such as those identified in Corollary 1 can be extended to larger arities (for example, by adding dummy arguments) and remain inexpressible. Hence our results refute this conjecture for all $k \geq 5$. However, we suggest that this conjecture can be refined to a similar statement concerning just those submodular functions which are expressible by binary submodular functions, as follows:

Conjecture 1. For all k, $\Gamma_{\mathsf{sub},k} \cap \langle \Gamma_{\mathsf{sub},2} \rangle = \mathrm{Cone}(\Gamma_{\mathsf{fans},k})$.

This conjecture was previously known to be true for $k \leq 3$ [29]; Theorem 1 shows that $\mathrm{Cone}(\Gamma_{\mathsf{fans},k}) \subseteq \Gamma_{\mathsf{sub},k} \cap \langle \Gamma_{\mathsf{sub},2} \rangle$ for all k, and Theorem 3 confirms that equality holds for $k = 4$.

3.4 The complexity of recognising expressible functions

Finally, we show that we can test efficiently whether a submodular polynomial of arity 4 is expressible by binary submodular polynomials.

Definition 6. *Let $p(x_1, x_2, x_3, x_4)$ be the polynomial representation of a 4-ary submodular cost function f. We denote by a_I the coefficient of the term $\prod_{i \in I} x_i$. We say that f satisfies condition* **Sep** *if for each $\{i, j\}, \{k, \ell\} \subset \{1, 2, 3, 4\}$, with i, j, k, ℓ distinct, we have $a_{\{i,j\}} + a_{\{k,\ell\}} + a_{\{i,j,k\}} + a_{\{i,j,\ell\}} \leq 0$.*

Theorem 4. *For any $f \in \Gamma_{\mathsf{sub},4}$, the following are equivalent:*

1. *$f \in \langle \Gamma_{\mathsf{sub},2} \rangle$.*
2. *f satisfies condition* **Sep**.

Proof. As in the proof of Theorem 3, we construct a set of 30 inequalities corresponding to the multimorphism $\mathcal{F}_{sep}$. Each of these inequalities on the values of a cost function can be translated into inequalities on the coefficients of the corresponding polynomial representation by a straightforward linear transformation. This calculation shows that 24 of the resulting inequalities impose the condition of submodularity, and the remaining 6 impose condition **Sep**. Hence a submodular cost function of arity 4 has the multimorphism $\mathcal{F}_{sep}$ if, and only if, its polynomial representation satisfies condition **Sep**. The result then follows from Theorem 3. □

Using Theorem 4, we can test whether optimisation problems given as a sum of submodular functions of arity 4 can be reduced to the MIN-CUT problem via the expressibility reduction. These problems arise in Computer Vision and in Valued Constraint Satisfaction Problems.

Furthermore, by Theorem 1, the number of extra variables needed in this reduction is rather small compared to the theoretical upper bound given in [4].

It is known that the problem of recognising whether an arbitrary degree-4 polynomial is submodular is co-NP-complete [9,15]. At the moment, the complexity of the recognition problem for submodular polynomials of degree 4 that are expressible by binary submodular polynomials is open.

References

1. Billionet, A., Minoux, M.: Maximizing a supermodular pseudo-Boolean function: a polynomial algorithm for cubic functions. Discrete Applied Math. **12** (1985) 1–11
2. Boros, E., Hammer, P.L.: Pseudo-Boolean optimization. Discrete Applied Mathematics **123**(1-3) (2002) 155–225
3. Burkard, R., Klinz, B., Rudolf, R.: Perspectives of Monge properties in optimization. Discrete Applied Mathematics **70** (1996) 95–161
4. Cohen, D., Cooper, M., Jeavons, P.: An algebraic characterisation of complexity for valued constraints. In: CP'06. Volume 4204 of LNCS. (2006) 107–121
5. Cohen, D., Cooper, M., Jeavons, P., Krokhin, A.: A maximal tractable class of soft constraints. Journal of Artificial Intelligence Research **22** (2004) 1–22
6. Cohen, D., Cooper, M., Jeavons, P., Krokhin, A.: Supermodular functions and the complexity of Max-CSP. Discrete Applied Mathematics **149** (2005) 53–72
7. Cohen, D., Cooper, M., Jeavons, P., Krokhin, A.: The complexity of soft constraint satisfaction. Artificial Intelligence **170** (2006) 983–1016
8. Cooper, M.: Minimization of Locally Defined Submodular Functions by Optimal Soft Arc Consistency. Constraints **13**(4):437–458, 2008.

9. Crama, Y.: Recognition problems for special classes of polynomials in 0-1 variables. Mathematical Programming **44** (1989) 139–155
10. Creignou, N., Khanna, S., Sudan, M.: Complexity Classification of Boolean Constraint Satisfaction Problems. SIAM (2001)
11. Deineko, V., Jonsson, P., Klasson, M., Krokhin, A.: The approximability of Max CSP with fixed-value constraints. Journal of the ACM **55**(4) (2008)
12. U. Feige, V. S. Mirrokni, J. Vondrák, Maximizing non-monotone submodular functions, in: Proceedings of FOCS'07, 2007.
13. Fisher, M., Nemhauser, G., Wolsey, L.: An analysis of approximations for maximizing submodular set functions-I. Mathematical Programming **14** (1978) 265–294
14. Fujishige, S.: Submodular Functions and Optimization. 2nd edn. Volume 58 of Annals of Discrete Mathematics. North-Holland, Amsterdam (2005)
15. Gallo, G., Simeone, B.: On the supermodular knapsack problem. Mathematical Programming **45** (1988) 295–309
16. Hammer, P.L.: Some network flow problems solved with pseudo-Boolean programming. Operations Research **13** (1965) 388–399
17. Iwata, S.: Submodular function minimization. Mathematical Programming **112** (2008) 45–64
18. Jeavons, P., Cohen, D., Cooper, M.: Constraints, consistency and closure. Artificial Intelligence **101**(1–2) (1998) 251–265
19. Jonsson, P., Klasson, M., Krokhin, A.: The approximability of three-valued MAX CSP. SIAM Journal on Computing **35**(6) (2006) 1329–1349
20. Kolmogorov, V., Zabih, R.: What energy functions can be minimized via graph cuts? IEEE Transactions on PAMI **26**(2) (2004) 147–159
21. Korte, B., Vygen, J.: Combinatorial Optimization. 4th edn. Volume 21 of Algorithms and Combinatorics. Springer-Verlag (2007)
22. Krokhin, A., Larose, B.: Maximizing supermodular functions on product lattices, with application to maximum constraint satisfaction. SIAM Journal on Discrete Mathematics **22**(1) (2008) 312–328
23. Lauritzen, S.L.: Graphical Models. Oxford University Press (1996)
24. Lovász, L.: Submodular functions and convexity. In Bachem, A., Grötschel, M., Korte, B., eds.: Math. Programming, Berlin, Springer-Verlag (1983) 235–257
25. Motzkin, T., Raiffa, H., Thompson, G., Thrall, R.: The double description method. Contributions to the Theory of Games. Vol. 2. Princeton Univ. Press (1953) 51–73
26. Narayanan, H.: Submodular Functions and Electrical Networks. (1997)
27. Nemhauser, G., Wolsey, L.: Integer and Combinatorial Optimization. (1988)
28. Orlin, J.B.: A faster strongly polynomial time algorithm for submodular function minimization. Mathematical Programming **118** (2009) 237–251
29. Promislow, S., Young, V.: Supermodular functions on finite lattices. Order **22**(4) (2005) 389–413
30. Rhys, J.: A selection problem of shared fixed costs and network flows. Management Science **17**(3) (1970) 200–207
31. Rossi, F., van Beek, P., Walsh, T.: The Handbook of Constraint Programming. (2006)
32. Schrijver, A.: Combinatorial Optimization: Polyhedra and Efficiency. Volume 24 of Algorithms and Combinatorics. Springer-Verlag (2003)
33. Topkis, D.: Supermodularity and Complementarity. (1998)
34. Zalesky, B.: Efficient determination of Gibbs estimators with submodular energy functions. arXiv:math/0304041v1 (February 2008)
35. Živný, S., Jeavons, P.G.: Classes of Submodular Constraints Expressible by Graph Cuts. In: CP'08. Volume 5202 of LNCS. (2008) 112–127