Classes of Submodular Constraints Expressible by Graph Cuts

Stanislav Živný and Peter G. Jeavons

Computing Laboratory, University of Oxford,
Wolfson Building, Parks Road, Oxford OX1 3QD, United Kingdom.
{stanislav.zivny,peter.jeavons}@comlab.ox.ac.uk

Abstract. Submodular constraints play an important role both in theory and practice of valued constraint satisfaction problems (VCSPs). It has previously been shown, using results from the theory of combinatorial optimisation, that instances of VCSPs with submodular constraints can be minimised in polynomial time. However, the general algorithm is of order $O(n^6)$ and hence rather impractical. In this paper, by using results from the theory of pseudo-Boolean optimisation, we identify several broad classes of submodular constraints over a Boolean domain which are expressible using binary submodular constraints, and hence can be minimised in cubic time. We also discuss the question of whether all submodular constraints of bounded arity over a Boolean domain are expressible using only binary submodular constraints, and can therefore be minimised efficiently.

1 Introduction

The Constraint Satisfaction Problem (CSP) is a general framework which can be used to model many different problems [11,17,21]. However, the CSP model considers only the feasibility of satisfying a collection of simultaneous requirements (so-called *hard constraints*).

Various extensions have been proposed to this model to allow it to deal with different kinds of optimisation criteria, or preferences, between different feasible solutions (so-called *soft constraints*). Two very general extended frameworks that have been proposed are the SCSP (semi-ring CSP) framework and the VCSP (valued CSP) framework [2]. The SCSP framework is slightly more general[1], but the VCSP framework is simpler, and yet sufficiently powerful to model a wide range of optimisation problems [2,21,22].

Informally, in the Valued Constraint Satisfaction Problem (VCSP) framework, an instance consists of a set of variables, a set of possible values, and a set of (soft) constraints. Each constraint has an associated cost function which assigns a cost (or a degree of violation) to every possible tuple of values for the

[1] The main difference is that costs in VCSPs represent violation levels and have to be totally ordered, whereas costs in SCSPs represent preferences and might be ordered only partially.

variables in the scope of the constraint. The goal is to find an assignment of values to all of the variables which has the minimum total cost. We remark that infinite costs can be used to indicate infeasible assignments (hard constraints), and hence the VCSP framework includes the standard CSP framework as a special case and is equivalent to the CONSTRAINT OPTIMISATION PROBLEM (COP) framework [21], which is widely used in practice.

One significant line of research on the VCSP is to identify restrictions which ensure that instances are solvable in polynomial time. There are two main types of restrictions that have been studied in the literature. Firstly, we can limit the *structure* of the instances. We will not deal with this approach in this paper.

Secondly, we can restrict the *forms* of the valued constraints which are allowed in the problem, giving rise to so-called *language restrictions*. Several language restrictions which ensure tractability have been identified in the literature, (see e.g., [8]). One important and well-studied restriction on valued constraints is *submodularity*. In fact the class of submodular constraints is the only non-trivial tractable case in the dichotomy classification of the Boolean VCSP [8].

The concept of submodularity not only plays an important role in theory, but is also very important in practice. For example, many of the problems that arise in computer vision can be expressed in terms of energy minimisation [16]. The problem of energy minimisation is NP-hard in general, and therefore a lot of research has been devoted to identifying instances which can be solved more efficiently. Kolmogorov and Zabih identified classes of instances for which the energy minimisation problem can be solved efficiently [16], and which are applicable to a wide variety of vision problems, including image restoration, stereo vision and motion tracking, image synthesis, image segmentation, multi-camera scene reconstruction and medical imaging. The so-called *regularity* condition, which specifies the efficiently solvable classes in [16], is equivalent to submodularity.

The notion of submodularity originally comes from combinatorial optimisation where submodular functions are defined on subsets of a given base set [14,18]. The time complexity of the fastest known algorithm for the problem of SUBMODULAR FUNCTION MINIMISATION (SFM) is roughly $O(n^6)$ [19]. However, there are several known special classes of SFM that can be solved more efficiently than the general case (see [3] for a survey).

Cohen et al. showed that VCSPs with submodular constraints over an arbitrary finite domain can be reduced to the SFM problem over a special family of sets known as a ring family [8]. This problem is equivalent to the general SFM problem [23], thus giving an algorithm of order $O(n^6 + n^5 L)$, where L is the look-up time (needed to evaluate an assignment to all variables), for any VCSP with submodular constraints. This tractability result has since been generalised to a wider class of valued constraints over arbitrary finite domains known as tournament-pair constraints [6]. An alternative approach can be found in [9].

In this paper we focus on submodular constraints over a Boolean domain $\{0, 1\}$, which correspond precisely to submodular set functions [8]. We describe an algorithm based on graph cuts which can be used to solve certain VCSPs with submodular constraints over a Boolean domain much more efficiently than the general case. Some of our results are closely related to known efficient cases of SFM, and other previous results from different areas of computer science, but we present them here in a unified and constraints-based framework which allows us to make the proofs more consistent and often simpler. Moreover, we explicitly discuss for the first time the *expressive power* of binary submodular constraints, and use this powerful idea in a consistent way to identify new classes of submodular constraints which can be solved efficiently.

The paper is organised as follows. In Section 2, we define the VCSP framework and submodular constraints, and note that submodular constraints over a Boolean domain can be represented by polynomials. In Section 3, we show that the standard (s,t)-MIN-CUT problem can be expressed in the VCSP framework with a restricted constraint language Γ_{cut}, and that any instance of VCSP(Γ_{cut}) is solvable in cubic time. Moreover, we show that Γ_{cut} can express all *binary* submodular constraints. In Section 4, we show that any instance of the VCSP with constraints whose corresponding polynomials have only non-positive coefficients for terms of degree ≥ 2 can be expressed in VCSP(Γ_{cut}). We show the same for all $\{0,1\}$-valued submodular constraints, and also for all *ternary* submodular constraints. In Section 5, we present a necessary condition for a quartic polynomial to be submodular. Moreover, for every $k \geq 4$, we identify new classes of k-ary submodular constraints which can be expressed over VCSP(Γ_{cut}), and thus solved efficiently. We then discuss the question of whether *all* submodular constraints of bounded arity can be expressed over Γ_{cut}. Finally, in Section 6, we summarise our work and discuss related and future work.

2 Definitions

2.1 Valued constraint satisfaction and expressibility

In this section we define the VALUED CONSTRAINT SATISFACTION PROBLEM (VCSP). In the original definition of this problem, given in [22], costs were allowed to lie in any positive totally ordered monoid called a *valuation structure*. For our purposes, it is sufficient to consider costs which lie in the set $\overline{\mathbb{Q}}_+$ consisting of all non-negative rational numbers together with infinity[2].

Given a fixed set D, a function from D^k to $\overline{\mathbb{Q}}_+$ will be called a *cost function*. If the range of ϕ is $\{0,\infty\}$, then ϕ is called a *crisp* cost function. Note that crisp cost functions correspond precisely to relations, so we shall use these terms interchangeably. If the range of ϕ lies entirely within $\mathbb{Q}_+$, the set of non-negative rationals, then ϕ is called a *finite-valued* cost function.

Definition 1. *An instance $\mathcal{P}$ of* VCSP *is a triple $\langle V, D, \mathcal{C}\rangle$, where V is a finite set of* variables*, which are to be assigned values from the set D, and $\mathcal{C}$ is a set of* valued constraints*. Each $c \in \mathcal{C}$ is a pair $c = \langle \sigma, \phi\rangle$, where σ is a tuple of variables of length $|\sigma|$, called the* scope *of c, and $\phi : D^{|\sigma|} \rightarrow \overline{\mathbb{Q}}_+$ is a cost function. An*

[2] See [10] for a discussion of why limiting ourselves to the $\overline{\mathbb{Q}}_+$ valuation structure is not a severe restriction.

assignment *for the instance $\mathcal{P}$ is a mapping s from V to D. The* cost *of an assignment s is defined as follows:*

$$Cost_{\mathcal{P}}(s) = \sum_{\langle\langle v_1,v_2,\ldots,v_m\rangle,\phi\rangle\in\mathcal{C}} \phi(\langle s(v_1), s(v_2), \ldots, s(v_m)\rangle).$$

A solution *to $\mathcal{P}$ is an assignment with minimum cost.*

Any set Γ of cost functions is called a *valued constraint language.* The class VCSP(Γ) is defined to be the class of all VCSP instances where the cost functions of all valued constraints lie in Γ.

In any VCSP instance, the variables listed in the scope of each valued constraint are explicitly constrained, in the sense that each possible combination of values for those variables is associated with a given cost. Moreover, if we choose *any* subset of the variables, then their values are constrained *implicitly* in the same way, due to the combined effect of the valued constraints. This motivates the concept of *expressibility* for cost functions, which is defined as follows:

Definition 2. *For any* VCSP *instance $\mathcal{I} = \langle V, D, \mathcal{C}\rangle$, and any list of variables of $\mathcal{I}$, $l = \langle v_1, \ldots, v_m\rangle$, the* projection *of $\mathcal{I}$ onto l, denoted $\pi_l(\mathcal{I})$, is the m-ary cost function defined as follows:*

$$\pi_l(\mathcal{I})(x_1, \ldots, x_m) \; = \min_{\{s:V\to D \mid \langle s(v_1),\ldots,s(v_m)\rangle = \langle x_1,\ldots,x_m\rangle\}} Cost_{\mathcal{I}}(s).$$

We say that a cost function ϕ is expressible *over a valued constraint language Γ if there exists an instance $\mathcal{I} \in$ VCSP(Γ) and a list l of variables of $\mathcal{I}$ such that $\pi_l(\mathcal{I}) = \phi$. We call the pair $\langle\mathcal{I}, l\rangle$ a* gadget *for expressing ϕ over Γ. Variables from $V \setminus l$ are called* extra *or* hidden *variables.*

Note that in the special case of relations (crisp cost functions) this notion of expressibility corresponds to the standard notion of expressibility using conjunction and existential quantification (*primitive positive formulas*) [4].

We denote by $\langle\Gamma\rangle$ the *expressive power* of Γ which is the set of all cost functions expressible over Γ up to additive and multiplicative constants.

2.2 Submodular functions and polynomials

A function $\psi : 2^V \to \mathbb{Q}$ defined on subsets of a set V is called a *submodular function* [18] if, for all subsets S and T of V, $\psi(S\cap T)+\psi(S\cup T) \leq \psi(S)+\psi(T)$. The problem of Submodular Function Minimisation (SFM) consists in finding a subset S of V for which the value of $\psi(S)$ is minimal.

For any lattice-ordered set D, a cost function $\phi : D^k \to \overline{\mathbb{Q}}_+$ is called *submodular* if for every $u, v \in D^k$, $\phi(\min(u, v)) + \phi(\max(u, v)) \leq \phi(u) + \phi(v)$ where both min and max are applied coordinate-wise on tuples u and v. Note that expressibility preserves submodularity: if every $\phi \in \Gamma$ is submodular, and $\phi' \in \langle\Gamma\rangle$, then ϕ' is also submodular.

Using results from [8] and [24], 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_{crisp}, that is, $\phi = \phi_{fin} + \phi_{crisp}$. Moreover, it is known that all submodular *relations* are binary decomposable [15], and hence expressible using only binary submodular relations. Therefore, when considering which cost functions are expressible over binary submodular cost functions, we can restrict our attention to *finite-valued* cost functions without any loss of generality.

In this paper we focus on problems over Boolean domains. We denote by $\mathit{\Gamma}_{\mathsf{sub,k}}$ the set of all finite-valued submodular cost functions of arity at most k on a Boolean domain $D = \{0, 1\}$, and we set $\mathit{\Gamma}_{\mathsf{sub}} = \bigcup_k \mathit{\Gamma}_{\mathsf{sub,k}}$. We will show below that VCSP($\mathit{\Gamma}_{\mathsf{sub,2}}$) can be solved in cubic time, and hence we will be concerned with what other cost functions are expressible over $\mathit{\Gamma}_{\mathsf{sub,2}}$, and so can also be solved efficiently.

A cost function of arity k can be represented as a table of values of size D^k. Alternatively, a (finite-valued) cost function $\phi : D^k \to \mathbb{Q}_+$ on a Boolean domain $D = \{0, 1\}$ can be uniquely represented as a *polynomial* in k (Boolean) variables with coefficients from $\mathbb{Q}$ [3] (such functions are sometimes called *pseudo-Boolean functions*). Hence, in what follows, we will often represent a finite-valued cost function on a Boolean domain by a polynomial.

Note that if $\mathit{\Gamma}$ is a set of cost functions on a Boolean domain, with arity at most k, then any instance of VCSP($\mathit{\Gamma}$) with n variables can be uniquely represented as a polynomial p in n Boolean variables, of degree at most k. Conversely, any such polynomial represents an n-ary cost function which can be expressed over a set of cost functions on a Boolean domain, with arity at most k. Note that $x^2 = x$, so p has at most 2^n terms which correspond to subsets of variables.

For polynomials over Boolean variables there is a standard way to define *derivatives* of each order (see [3]). For example, the second order derivative of a polynomial p, with respect to the first two indices, denoted $\delta_{1,2}(\boldsymbol{x})$, is defined as $p(1, 1, \boldsymbol{x}) - p(1, 0, \boldsymbol{x}) - p(0, 1, \boldsymbol{x}) + p(0, 0, \boldsymbol{x})$. Analogously for all other pairs of indices.

Proposition 3 ([3]). *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 $\delta_{i,j}(\boldsymbol{x})$ are non-positive for all $1 \le i < j \le n$ and all $\boldsymbol{x} \in D^{n-2}$.*

Corollary 4. *A* quadratic *polynomial $a_0 + \sum_{i=1}^{n} a_i x_i + \sum_{1 \le i < j \le n} a_{ij} x_i x_j$ over Boolean variables $x_1, \ldots, x_k$, represents a submodular cost function if and only if $a_{ij} \le 0$ for every $1 \le i < j \le n$.*

3 Binary submodular constraints

In this section we show that a constraint language $\mathit{\Gamma}_{\mathsf{cut}}$, consisting of certain simple binary and unary cost functions over a Boolean domain, has cubic time complexity. We also show that $\mathit{\Gamma}_{\mathsf{cut}}$ can express any binary submodular cost function over a Boolean domain, that is, $\mathit{\Gamma}_{\mathsf{sub,2}} \subseteq \langle \mathit{\Gamma}_{\mathsf{cut}} \rangle$. It follows that any instance of VCSP($\mathit{\Gamma}_{\mathsf{sub,2}}$) can also be solved in cubic time.

For any $w \in \overline{\mathbb{Q}}_+$, we define the binary cost function χ^w as follows:

$$\chi^w(x, y) \;=\; \begin{cases} w & \text{if } (x, y) = (0, 1), \\ 0 & \text{otherwise.} \end{cases}$$

For any $d \in D$ and $c \in \overline{\mathbb{Q}}_+$, we define the unary cost function μ_d^c as follows:

$$\mu_d^c = \begin{cases} c & \text{if } x \neq d, \\ 0 & \text{if } x = d. \end{cases}$$

It is straightforward to check that all χ^w and μ_d^c are submodular.

We define the constraint language Γ_{cut} to be the set of all cost functions χ^w and μ_d^c over a Boolean domain, for $c, w \in \overline{\mathbb{Q}}_+$ and $d \in \{0, 1\}$.

Theorem 5. *The problems* (s,t)-Min-Cut *and* VCSP(Γ_{cut}) *are linear-time equivalent.*

Proof. Consider any instance of (s,t)-Min-Cut with (directed) graph $G = \langle V, E \rangle$ and weight function $w : E \to \overline{\mathbb{Q}}_+$. Define a corresponding instance $\mathcal{I}$ of VCSP(Γ_{cut}) as follows:

$$\mathcal{I} = \langle V, \{0,1\}, \{\langle\langle i,j \rangle, \chi^{w(i,j)}\rangle \mid \langle i,j \rangle \in E\} \cup \{\langle s, \mu_0^\infty \rangle, \langle t, \mu_1^\infty \rangle\}\rangle.$$

Note that in any solution to $\mathcal{I}$ the source and target nodes, s and t, must take the values 0 and 1, respectively. Moreover, the weight of any cut containing s and not containing t is equal to the cost of the corresponding assignment to $\mathcal{I}$. Hence we have shown that (s,t)-Min-Cut can be reduced to VCSP(Γ_{cut}) in linear time.

On the other hand, given an instance $\mathcal{I} = \langle V, D, \mathcal{C} \rangle$ of VCSP(Γ_{cut}), construct a graph on $V \cup \{s, t\}$ as follows: any unary constraint on variable v with cost function μ_0^c (respectively μ_1^c) is represented by an edge of weight c from the source node s to node v (respectively, from node v to the target node t). Any binary constraint on variables $\langle v_1, v_2 \rangle$ with cost function χ^w is represented by an edge of weight w from node v_1 to v_2. It is straightforward to check that a solution to $\mathcal{I}$ corresponds to a minimum (s,t)-cut of this graph. $\square$

Corollary 6. VCSP(Γ_{cut}) *can be solved in cubic time.*

Proof. By Theorem 5, VCSP(Γ_{cut}) has the same time complexity as (s,t)-Min-Cut, which is known to be solvable in cubic time [13]. $\square$

Using a standard reduction (see, for example, [3]), we now show that all *binary* submodular cost functions over a Boolean domain can be expressed over Γ_{cut}.

Theorem 7. $\Gamma_{\mathsf{sub},2} \subseteq \langle \Gamma_{\mathsf{cut}} \rangle$.

Proof. By Corollary 4, any cost function from $\Gamma_{\mathsf{sub},2}$ can be represented by a quadratic Boolean polynomial $p(x_1, x_2) = a_0 + a_1x_1 + a_2x_2 + a_{12}x_1x_2$ where $a_{12} \leq 0$. This can then be re-written as

$$p(x_1, x_2) = a'_0 + \sum_{i \in P} a'_i x_i + \sum_{j \in N} a'_j (1 - x_j) + a'_{12}(1 - x_1)x_2,$$

where $P \cap N = \emptyset$, $P \cup N = \{1, 2\}$, $a'_{12} = -a_{12}$, and $a'_i, a'_j, a'_{12} \geq 0$. (This is known as a *posiform* [3].)

Hence p can be expressed over Γ_{cut} (up to the constant a'_0) by the gadget $\langle \mathcal{I}, \langle x_1, x_2 \rangle \rangle$, where $\mathcal{I}$ is the instance $\langle \{x_1, x_2, s, t\}, \{0,1\}, \mathcal{C} \rangle$ of VCSP(Γ_{cut}) and

$$\mathcal{C} = \{\langle \langle s, x_i \rangle, \chi^{a'_i} \rangle \mid i \in P\} \cup \{\langle \langle x_j, t \rangle, \chi^{a'_j} \rangle \mid j \in N\}$$
$$\cup \{\langle \langle s \rangle, \mu_0^\infty \rangle, \langle \langle t \rangle, \mu_1^\infty \rangle, \langle \langle x_1, x_2 \rangle, \chi^{a'_{12}} \rangle \}.$$

□

Corollary 8. VCSP($\Gamma_{\mathsf{sub,2}}$) *can be solved in cubic time.*

Proof. By Theorem 7, any instance of VCSP($\Gamma_{\mathsf{sub,2}}$) can be reduced to VCSP(Γ_{cut}) in linear time by replacing each constraint with a suitable gadget of fixed size. The result then follows from Corollary 6. (Note that we can use the same vertices s and t for all constraints.) □

4 Negative higher degree terms, $\{0,1\}$-valued and ternary submodular constraints

In this section we extend the results from Section 3 to three further classes of constraints over a Boolean domain: submodular constraints whose corresponding polynomials have negative coefficients for all terms of degree ≥ 2; $\{0,1\}$-valued submodular constraints; and ternary submodular constraints. We show that the cost functions for these three classes of submodular constraints can all be expressed over $\Gamma_{\mathsf{sub,2}}$, and hence can be minimised in cubic time in the number of variables plus the number of higher-order (non-binary) constraints.

Define $\Gamma_{\mathsf{neg,k}}$ to be the set of all cost functions over a Boolean domain, of arity at most k, whose corresponding polynomials have negative coefficients for all terms of degree greater than or equal to 2. It is easy to check that these cost functions, sometimes called *negative-positive*, are submodular. Set $\Gamma_{\mathsf{neg}} = \bigcup_k \Gamma_{\mathsf{neg,k}}$. The minimisation of cost functions chosen from Γ_{neg} using min-cuts was first studied in [20].

Theorem 9. $\Gamma_{\mathsf{neg}} \subseteq \langle \Gamma_{\mathsf{sub,2}} \rangle$.

Proof. Consider the following polynomial:

$$p_0(x_1, \ldots, x_n) = \min_{y \in \{0,1\}} \{-y + y \sum_{i \in A} (1 - x_i)\}.$$

It is straightforward to check that for a given $A \subseteq \{1, \ldots, n\}$, $p_0(\boldsymbol{x}) = -1$ if $A \subseteq \boldsymbol{x}$ and $p_0(\boldsymbol{x}) = 0$ otherwise (where $A \subseteq \boldsymbol{x}$ means $\forall i \in A, x_i = 1$).

Now, given any polynomial of the form $p_1(x_1, \ldots, x_n) = -a_{klm} x_k x_l x_m + Q$, where Q consists of terms of degree ≤ 2, we can use a similar construction to p_0 to obtain

$$p_1(x_1, \ldots, x_n) = \min_{y \in \{0,1\}} \{Q + a_{klm}(-y + y \sum_{i \in \{k,l,m\}} (1 - x_i))\}.$$

Given any polynomial p, we can use a similar construction to replace each term of degree ≥ 3 in turn, introducing a distinct new variable y each time.

Proceeding in this way, we can express any polynomial p representing a cost function in Γ_{neg} as a quadratic polynomial with non-positive quadratic coefficients, introducing k new variables, where k is the total number of terms of degree ≥ 3. Such a quadratic polynomial can be expressed over $\Gamma_{\mathsf{sub},2}$, by Corollary 4. $\square$

Corollary 10. *For any fixed k,* VCSP($\Gamma_{\mathsf{neg,k}}$) *can be solved in cubic time.*

Proof. By Theorem 9, any instance of VCSP($\Gamma_{\mathsf{neg,k}}$) can be reduced to VCSP($\Gamma_{\mathsf{sub},2}$) in linear time by replacing each constraint with a suitable gadget. For any fixed k, the number of new variables introduced in any of these gadgets is bounded by a constant. The result then follows from Corollary 8.

Next we consider the class of submodular constraints over a Boolean domain which take only the cost values 0 and 1. (Such constraints can be used to model optimisation problems such as Max-CSP, see [7].) Define $\Gamma_{\{0,1\},\mathsf{k}}$ to be the set of all $\{0,1\}$-valued submodular cost functions over a Boolean domain, of arity at most k, and set $\Gamma_{\{0,1\}} = \cup_k \Gamma_{\{0,1\},\mathsf{k}}$. The minimisation of submodular cost functions from $\Gamma_{\{0,1\}}$ was studied in [11], where they were called *2-monotone* functions. The equivalence of 2-monotone and submodular cost functions and a generalisation of 2-monotone functions to non-Boolean domains was shown in [7].

Definition 11. *A cost function ϕ is called* 2-monotone *if there exist two sets $A, B \subseteq \{1, \ldots, n\}$ such that $\phi(\boldsymbol{x}) = 0$ if $A \subseteq \boldsymbol{x}$ or $\boldsymbol{x} \subseteq B$ and $\phi(\boldsymbol{x}) = 1$ otherwise (where $A \subseteq \boldsymbol{x}$ means $\forall i \in A, x_i = 1$ and $\boldsymbol{x} \subseteq B$ means $\forall i \notin B, x_i = 0$).*

Theorem 12. $\Gamma_{\{0,1\}} \subseteq \langle \Gamma_{\mathsf{sub},2} \rangle$.

Proof. Any 2-monotone cost function ϕ can be expressed over $\Gamma_{\mathsf{sub},2}$ using 2 extra variables, y_1, y_2:

$$\phi(\boldsymbol{x}) = \min_{y_1, y_2 \in \{0,1\}} \{(1 - y_1)y_2 + y_1 \sum_{i \in A} (1 - x_i) + (1 - y_2) \sum_{i \notin B} x_i\}.$$

$\square$

Corollary 13. *For any fixed k,* VCSP($\Gamma_{\{0,1\},\mathsf{k}}$) *can be solved in cubic time.*

Finally, we consider the class $\Gamma_{\mathsf{sub},3}$ of ternary submodular cost functions over a Boolean domain. This class was studied in [1], from where we obtain the following useful characterisation of cubic submodular polynomials.

Lemma 14 ([1]). *A cubic polynomial $p(x_1, \ldots, x_n)$ over Boolean variables represents a submodular cost function if and only if it can be written as*

$$p(x_1, \ldots, x_n) = a_0 + \sum_{\{i\} \in C_1^+} a_i x_i - \sum_{\{i\} \in C_1^-} a_i x_i - \sum_{\{i,j\} \in C_2} a_{ij} x_i x_j + \sum_{\{i,j,k\} \in C_3^+} a_{ijk} x_i x_j x_k - \sum_{\{i,j,k\} \in C_3^-} a_{ijk} x_i x_j x_k,$$

where

1. $a_i, a_{ij}, a_{ijk} \geq 0$ $(\{i\} \in C_1^+ \cup C_1^-, \{i,j\} \in C_2, \{i,j,k\} \in C_3^+ \cup C_3^-)$,
2. $\forall \{i,j\} \in C_2,\ a_{ij} + \sum_{k|\{i,j,k\} \in C_3^+} a_{ijk} \leq 0$.

Theorem 15. $\Gamma_{\mathsf{sub},3} \subseteq \langle \Gamma_{\mathsf{sub},2} \rangle$.

Proof. Let p be a polynomial representing an arbitrary cost function in $\Gamma_{\mathsf{sub},3}$. By Lemma 14, the quadratic terms in p are non-positive. We already know how to express a negative cubic term using a gadget over $\Gamma_{\mathsf{sub},2}$ (Theorem 9). To express a positive cubic term, consider the following identity:

$$x_i x_j x_k - x_i x_j - x_i x_k - x_j x_k = \min_{y \in \{0,1\}} \{(1 - x_i - x_j - x_k) y\}.$$

We can replace a positive cubic term $a_{ijk} x_i x_j x_k$ in p with

$$\min_{y \in \{0,1\}} \{a_{ijk}(1 - x_i - x_j - x_k) y + a_{ijk}(x_i x_j + x_i x_k + x_j x_k)\}.$$

It remains to check that all quadratic coefficients of the resulting polynomial are non-positive. However, this is ensured by the second condition from Lemma 14. □

Corollary 16. VCSP$(\Gamma_{\mathsf{sub},3})$ *can be solved in cubic time.*

5 Submodular constraints of arity 4 and higher

In this section we investigate the question of which submodular constraints of arity 4 or higher can be expressed by binary submodular constraints. We derive a necessary condition for a 4-ary constraint over a Boolean domain to be submodular. We also present some sufficient conditions, which give rise to new classes of submodular constraints which can be expressed over $\Gamma_{\mathsf{sub},2}$, and hence minimised efficiently. First, we prove the sufficient condition for 4-ary submodular cost functions. Next, we generalise it to k-ary submodular cost functions for every $k \geq 4$. Finally, we discuss the general question of which submodular cost functions over a Boolean domain can be expressed with binary submodular cost functions.

5.1 4-ary constraints

One might hope to obtain a nice characterisation of 4-ary submodular cost functions over a Boolean domain similar to Lemma 14. However, it has been shown that testing whether a given quartic Boolean polynomial is submodular is co-NP-complete [12]. Hence, one is unlikely to find a "simple" characterisation; any characterisation is likely to involve an exponential blow-up (e.g., quantification over a non-constant number of variables). However, we can obtain the following necessary condition.

Lemma 17. *If a quartic polynomial $p(x_1, \dots, x_n)$ over Boolean variables represents a submodular cost function, then it can be written such that, for all $\{i,j\} \in C_2$:*

1. $a_{ij} \leq 0$, *and*
2. $a_{ij} + \sum_{k|\{i,j,k\} \in C_3^+} a_{ijk} + \sum_{k,l|\{i,j,k,l\} \in C_4^+} a_{ijkl} + F_{ij} \leq 0$

where $F_{i,j}$ is a non-positive value which is equal to the sum of the coefficients of certain non-positive cubic and quartic terms, C_2 denotes the set of quadratic terms, and C_i^+ denotes the set of terms of degree i with positive coefficients, for $i = 3, 4$.

Proof. Let p be a quartic submodular polynomial and let i and j be given, then p can always be put in a form so that the second order derivative is:

$$\begin{aligned}\delta_{i,j} = a_{i,j} + \sum_{k|\{i,j,k\} \in C_3^+} a_{ijk} x_k + \sum_{k,l|\{i,j,k,l\} \in C_4^+} a_{ijkl} x_k x_l \\ - \sum_{k|\{i,j,k\} \in C_3^-} a_{ijk} x_k - \sum_{k,l|\{i,j,k,l\} \in C_4^-} a_{ijkl} x_k x_l.\end{aligned}$$

Consider an assignment which sets $x_i = x_j = 1$ and $x_k = 0$ $\forall k \neq i \neq j$. By Proposition 3, $a_{ij} \leq 0$, which proves the first condition. By setting $x_k = 1$ for all k such that $\{i,j,k\} \in C_3^+$ and $x_k = x_l = 1$ for all k, l such that $\{i,j,k,l\} \in C_4^+$, we get the second condition. We set to 1 all variables which occur in some positive cubic or quartic term. The second condition then says that the sum of all these positive coefficients minus those which are forced, by our setting of variables, to be 1 (F_{ij}), is at most 0. (Note that this also proves Lemma 14.) □

Next we show a useful example of a 4-ary submodular cost function which can be expressed over the binary submodular cost functions using one extra variable.

Example 18. Let ϕ be the 4-ary cost function defined as follows: $\phi(\boldsymbol{x}) = \min\{2k, 5\}$, where k is the number of 0s in $\boldsymbol{x} \in \{0,1\}^4$. The corresponding quartic polynomial representing ϕ is

$$p(x_1, x_2, x_3, x_4) = 5 + x_1x_2x_3x_4 - x_1x_2 - x_1x_3 - x_1x_4 - x_2x_3 - x_2x_4 - x_3x_4.$$

Is is easy to check that p is submodular. It can be shown by simple case analysis that p cannot be expressed as a quadratic polynomial with non-positive quadratic coefficients (from the definition of p, the polynomial would have to be $5 - x_1x_2 - x_1x_3 - x_1x_4 - x_2x_3 - x_2x_4 - x_3x_4$ which is not equal to p on $x_1 = x_2 = x_3 = x_4 = 1$).

However, p can be expressed over $\Gamma_{\mathsf{sub},2}$ using just one extra variable, via the following gadget:

$$p(x_1, x_2, x_3, x_4) = \min_{y \in \{0,1\}} \{5 + (3 - 2x_1 - 2x_2 - 2x_3 - 2x_4)y\}.$$

Using the same notation as in Lemma 17, define $\Gamma_{\mathsf{new},4}$ to be the set of all 4-ary submodular cost functions over a Boolean domain whose corresponding quartic polynomials satisfy, for every $i < j$,

$$a_{ij} + \sum_{k|\{i,j,k\}\in C_3^+} a_{ijk} + \sum_{k,l|\{i,j,k,l\}\in C_4^+} a_{ijkl} \quad \leq \quad 0. \tag{1}$$

Theorem 19. $\Gamma_{\mathsf{new},4} \subseteq \langle \Gamma_{\mathsf{sub},2} \rangle$.

Proof. Let $\phi \in \Gamma_{\mathsf{new},4}$ and let p be the corresponding polynomial which represents ϕ. First, replace all negative cubic and quartic terms using the construction in Theorem 9. As in the proof of Theorem 15, replace every positive cubic term $a_{ijk}x_ix_jx_k$ in p with

$$\min_{y\in\{0,1\}} \{a_{ijk}(1 - x_i - x_j - x_k)y + a_{ijk}(x_ix_j + x_ix_k + x_jx_k)\}.$$

Using the same construction as in Example 18, replace every positive quartic term $a_{ijkl}x_ix_jx_kx_l$ with

$$\min_{y\in\{0,1\}} \{a_{ijkl}(3 - 2x_i - 2x_j - 2x_k - 2x_l)y \\ + a_{ijkl}(x_ix_j + x_ix_k + x_ix_l + x_jx_k + x_jx_l + x_kx_l)\}.$$

It only remains to check that all quadratic coefficients in the resulting polynomial are non-positive. However, this is ensured by the definition of $\Gamma_{\mathsf{new},4}$ and by the choice of the gadgets. □

Corollary 20. VCSP($\Gamma_{\mathsf{new},4}$) *can be solved in cubic time.*

Unfortunately, our next example shows that $\Gamma_{\mathsf{new},4} \subsetneq \Gamma_{\mathsf{sub},4}$, and it remains an open question whether *all* 4-ary submodular constraints over a Boolean domain can be expressed over $\Gamma_{\mathsf{sub},2}$.

Example 21. Define a 4-ary submodular cost function ϕ as follows: $\phi(\boldsymbol{x}) = \min(3k, 7) + 2y + z$, where k is the number of 0s in $\boldsymbol{x} \in \{0,1\}^4$, $y = 1$ if and only if $\boldsymbol{x} = \langle 1,1,1,0\rangle$, and $z = 1$ if and only if $\boldsymbol{x} = \langle 1,1,0,0\rangle$. The corresponding polynomial representing ϕ is

$$p(x_1,x_2,x_3,x_4) = 7 + 2x_1x_2x_3x_4 - 2x_1x_2x_4 - x_1x_3x_4 - x_2x_3x_4 \\ - x_1x_3 - x_1x_4 - x_2x_3 - x_2x_4 - x_3x_4.$$

It is easy to check that ϕ is submodular, but $\phi \notin \Gamma_{\mathsf{new},4}$ (e.g., for $i = 1$ and $j = 2$, the expression in Equation 1 gives 2), so Theorem 19 does not apply.

As in Example 18, by a simple case analysis, it can be shown that ϕ cannot be expressed over $\Gamma_{\mathsf{sub},2}$ without extra variables. However, the following gadget shows that ϕ is in fact expressible over $\Gamma_{\mathsf{sub},2}$ using just two extra variables:

$$p(x_1,x_2,x_3,x_4) = 7 - x_1x_4 - x_2x_4 - x_3x_4 \\ + \min_{y_1,y_2\in\{0,1\}} \{2y_1 + 3y_2 - y_1y_2 - y_1(x_1 + x_2 + 2x_3) - y_2(x_1 + x_2 + 2x_4)\}.$$

5.2 The general case

We now generalise the result from the previous section to subclasses of submodular constraints of arbitrary arities. We define $\Gamma_{\mathsf{new,k}}$ to be the set of all k-ary submodular cost functions over a Boolean domain whose corresponding polynomials satisfy, for every $1 \leq i < j \leq k$,

$$a_{ij} + \sum_{s=1}^{k-2} \sum_{\{i,j,i_1,\ldots,i_s\} \in C^+_{s+2}} a_{i,j,i_1,\ldots,i_s} \quad \leq \quad 0.$$

In other words, for any $1 \leq i < j \leq k$, the sum of a_{ij} and all positive coefficients of cubic and higher degree terms which include x_i and x_j is non-positive.

Theorem 22. *For every $k \geq 4$, $\Gamma_{\mathsf{new,k}} \subseteq \langle \Gamma_{\mathsf{sub,2}} \rangle$.*

Proof. Note that the case $k = 4$ is proved by Theorem 19. First we show that in order to prove the statement of the theorem, it is sufficient to have a uniform way of generating gadgets over $\Gamma_{\mathsf{sub,2}}$ for polynomials of the following type:

$$p_k(x_1, \ldots, x_k) = \prod_{i=1}^{k} x_i - \sum_{1 \leq i < j \leq k} x_i x_j.$$

Note that $p_k(\boldsymbol{x}) = -\binom{m}{2}$, where m is the number of 1s in $\boldsymbol{x}$, and $\binom{0}{2} = \binom{1}{2} = 0$, unless $m = k$ ($\boldsymbol{x}$ consists of 1s only), in which case $p_k(\boldsymbol{x}) = -\binom{m}{2} + 1$.

Assume that for any $k \geq 5$, we can construct a gadget $\mathcal{P}_k$ for p_k over $\Gamma_{\mathsf{sub,2}}$. Given a cost function $\phi \in \Gamma_{\mathsf{new,k}}$, let p be the corresponding polynomial which represents ϕ. By the construction in Theorem 9, we can replace all negative terms of degree ≥ 3. By the constructions in Theorem 15 and Theorem 19, we can replace all positive cubic and quartic terms. Now for any positive term of degree d, $5 \leq d \leq k$, we replace it with the gadget $\mathcal{P}_d$. This construction works if all quadratic coefficients of the resulting polynomial are non-positive. However, this is ensured by the definition of $\Gamma_{\mathsf{new,k}}$ and by the choice of the gadgets.

It remains to show how to uniformly generate gadgets for p_k, where $k \geq 5$. We claim, that for any $k \geq 4$, the following, denoted by $\mathcal{P}_k$, is a gadget for p_k:

$$p_k(x_1, \ldots, x_k) = \min_{y_0, \ldots, y_{k-4} \in \{0,1\}} \{ y_0 (3 - 2 \sum_{i=1}^{k} x_i) + \sum_{j=1}^{k-4} y_j (2 + j - \sum_{i=1}^{k} x_i) \}.$$

Notice that in the case of $k = 4$, the gadget corresponds to the gadget used in the proof of Theorem 19, and therefore the base case is proved. We proceed by induction in k. Assume that $\mathcal{P}_i$ is a gadget for p_i for every $i \leq k$. We prove that $\mathcal{P}_{k+1}$ is a gadget for p_{k+1}.

Firstly, take the gadget $\mathcal{P}_k$ for p_k, and replace every sum $\sum_{i=1}^{k} x_i$ with $\sum_{i=1}^{k+1} x_i$. We denote the new gadget $\mathcal{P}'$. It is not difficult to see that $\mathcal{P}'$ is a valid gadget for p_{k+1} on all assignments with at most $k - 1$ 1s. Also, on any assignment with exactly k 1s, $\mathcal{P}'$ returns $-\binom{k}{2}+1$. On the assignment consisting of 1s only, $\mathcal{P}'$ returns: $-\binom{k}{2}+1-2-1(k-4)$. This can be simplified as follows: $-\binom{k}{2}+1-2-k+4=-\binom{k}{2}+1-k+2=-(\binom{k}{2}+\binom{k}{1})+1+2=-\binom{k+1}{2}+1+2$. Hence $\mathcal{P}'$ is *almost* a gadget for p_{k+1}: we only need to subtract 1 on an assignment which has exactly k 1s, and subtract 2 on the assignment consisting of 1s only. But this is exactly what $\min_{y_{k-3}\in\{0,1\}}\{y_{k-3}(2+(k-3)-\sum_{i=1}^{k+1}x_i)\}$ does. Therefore, we have established that $\mathcal{P}_{k+1}$ is a gadget for p_{k+1}, which finishes the proof of the theorem. □

Corollary 23. *For any $k \geq 4$, $\mathrm{VCSP}(\Gamma_{\mathsf{new,k}})$ can be solved in cubic time.*

The general question we are investigating is what can be expressed over $\Gamma_{\mathsf{sub,2}}$. Denote by $\langle\Gamma_{\mathsf{sub,2}}\rangle^{\mathsf{m}}$ the set of all (submodular) cost functions expressible over $\Gamma_{\mathsf{sub,2}}$ with at most m extra variables. Clearly, $\langle\Gamma_{\mathsf{sub,2}}\rangle^{\mathsf{m}} \subseteq \langle\Gamma_{\mathsf{sub,2}}\rangle^{\mathsf{m+1}}$ for every $m \geq 0$.

In the proof of Theorem 22, we proved that, for any $k \geq 4$, $\Gamma_{\mathsf{new,k}} \subseteq \langle\Gamma_{\mathsf{sub,2}}\rangle^{m}$ where $m = k-3$. We have since found out that a slightly stronger result was obtained independently by Zalesky. He has shown that $\Gamma_{\mathsf{new,k}} \subseteq \langle\Gamma_{\mathsf{sub,2}}\rangle^{\mathsf{m}}$ where $m = \lfloor\frac{k-1}{2}\rfloor$ (see the unpublished manuscript [25]). This result yields the same cubic time complexity for $\mathrm{VCSP}(\Gamma_{\mathsf{new,k}})$.

We saw in Section 5 that $\langle\Gamma_{\mathsf{sub,2}}\rangle^{1}$ is strictly larger than $\langle\Gamma_{\mathsf{sub,2}}\rangle^{0}$ (see Example 18). In other words, allowing a single hidden variable strictly increases the expressive power of $\Gamma_{\mathsf{sub,2}}$. On the other hand, we do not know whether allowing further hidden variables increases the expressive power any further. In other words, it is an open question whether $\langle\Gamma_{\mathsf{sub,2}}\rangle^{\mathsf{m}} \subsetneq \langle\Gamma_{\mathsf{sub,2}}\rangle^{\mathsf{m+1}}$ for any $m \geq 1$. We suspect that some of these inclusions are strict (see Example 21), as we carried out a computer-assisted search for gadgets, using the constraint-solver MINION[3], and for some cost function we were not able to find a gadget with a given number of extra variables.

However, we do know that there is a limit to the additional expressive power that can be gained by allowing an arbitrary number of hidden variables. This is a consequence of the following result, which is a general result about expressibility, and not specific to submodular constraints or Boolean domains.

Proposition 24. *If a cost function $\phi : D^k \to \overline{\mathbb{Q}}_+$ is expressible over Γ, then ϕ is expressible over Γ using at most $|D|^{|D|^k}$ hidden variables.*

Proof. If $\phi \in \langle\Gamma\rangle$, then by Definition 2, there is a gadget $\langle\mathcal{P}, l\rangle$, where $l = \langle v_1,\ldots,v_k\rangle$, for expressing ϕ over Γ. For the gadget $\langle\mathcal{P}, l\rangle$ to express ϕ, it has to define ϕ on each of the $|D|^k$ different assignments to l. Let each of these $|D|^k$ assignments be extended to a complete assignment to all variables of $\mathcal{P}$ (including hidden variables) in a way that minimises the total cost. For each hidden variable v of $\langle\mathcal{P}, l\rangle$, we can use the list of $|D|^k$ values assigned to v by these complete assignments to label the variable v. If there are more then $|D|^{|D|^k}$ hidden variables, then two of them will receive the same label. However,

[3] Available from `http://minion.sourceforge.net/`

this implies that one of the two is redundant, as all constraints involving that variable can replace it with the other variable without changing the overall cost. Hence we require at most $|D|^{|D|^k}$ distinct hidden variables to express ϕ. □

6 Conclusion

In this paper we first considered binary submodular constraints over a Boolean domain, and showed that they can be minimised in cubic time via a reduction to the minimum cut problem for graphs. We then investigated which other submodular constraints are *expressible* using binary submodular constraints over a Boolean domain, and hence can also be minimised efficiently using minimum cuts.

Using known results from combinatorial optimisation, we identified several such classes of constraints, including all ternary submodular constraints, and all $\{0,1\}$-valued submodular constraints of any arity. By constructing suitable gadgets, we identified certain new classes of k-ary submodular constraints, where $k \geq 4$, which can also be expressed by binary submodular constraints.

The main open problem raised by this paper is whether *all* bounded-arity submodular constraints over a Boolean domain can be expressed by binary submodular constraints, and hence solved in cubic time. In terms of polynomials, this is equivalent to the following problem: can any Boolean polynomial with non-positive second order derivatives be expressed as the projection of a quadratic polynomial with non-positive quadratic coefficients?

The results presented in this paper provide a partial answer to this question using constructive methods which can be used to obtain concrete reductions to problems such as (s,t)-MIN-CUT. We note that an alternative general approach to the problem of determining the expressive power of valued constraints was developed in [5]. It was shown there that the expressive power of any valued constraint language is characterised by a collection of algebraic properties called *fractional polymorphisms* [5]. In order to show that $\Gamma_{\mathsf{sub,k}} \subseteq \langle \Gamma_{\mathsf{sub,2}} \rangle$ it would therefore be sufficient to show that $\Gamma_{\mathsf{sub,2}}$ and $\Gamma_{\mathsf{sub,k}}$ have the same fractional polymorphisms. However, this algebraic approach is *non-constructive*, and hence has certain limitations: even if it could be established in this way that $\Gamma_{\mathsf{sub,k}} \subseteq \langle \Gamma_{\mathsf{sub,2}} \rangle$, this would not directly provide us with a gadget for any given problem (and hence an efficient algorithm). Conversely, if it could be established using the algebraic approach that $\Gamma_{\mathsf{sub,k}} \not\subseteq \langle \Gamma_{\mathsf{sub,2}} \rangle$, that would still leave open the question of identifying which subclasses of $\Gamma_{\mathsf{sub,k}}$ *can* be expressed over $\Gamma_{\mathsf{sub,2}}$, and hence solved efficiently. This paper provides a first step in answering that question using constructive techniques that could be implemented in valued constraint solvers.

Acknowledgements The authors would like to thank David Cohen and Martin Cooper for fruitful discussions on submodular constraints and Chris Jefferson for help with using the constraint-solver MINION, which helped us to find and simplify some of the gadgets presented in this paper. Stanislav Živný gratefully acknowledges the support of EPSRC grant EP/F01161X/1.

References

1. Billionet, A., Minoux, M.: Maximizing a supermodular pseudo-boolean function: a polynomial algorithm for cubic functions. D. App. Mathematics **12** (1985) 1–11
2. Bistarelli, S., Fargier, H., Montanari, U., Rossi, F., Schiex, T., Verfaillie, G.: Semiring-based CSPs and valued CSPs: Frameworks, properties, and comparison. Constraints **4** (1999) 199–240
3. Boros, E., Hammer, P.L.: Pseudo-boolean optimization. Discrete Applied Mathematics **123**(1-3) (2002) 155–225
4. Bulatov, A., Krokhin, A., Jeavons, P.: Classifying the complexity of constraints using finite algebras. SIAM Journal on Computing **34**(3) (2005) 720–742
5. Cohen, D., Cooper, M., Jeavons, P.: An algebraic characterisation of complexity for valued constraints. In: CP'06. Volume 4204 of LNCS. (2006) 107–121
6. Cohen, D., Cooper, M., Jeavons, P.: Generalising submodularity and Horn clauses: Tractable optimization problems defined by tournament pair multimorphisms. Theoretical Computer Science (2008) (in press).
7. Cohen, D., Cooper, M., Jeavons, P., Krokhin, A.: Supermodular functions and the complexity of Max-CSP. Discrete Applied Mathematics **149** (2005) 53–72
8. Cohen, D., Cooper, M., Jeavons, P., Krokhin, A.: The complexity of soft constraint satisfaction. Artificial Intelligence **170** (2006) 983–1016
9. Cooper, M.C.: Minimization of locally defined submodular functions by optimal soft arc consistency. Constraints **13** (2008)
10. Cooper, M.: High-order consistency in valued constraint satisfaction. Constraints **10** (2005) 283–305
11. Creignou, N., Khanna, S., Sudan, M.: Complexity Classification of Boolean Constraint Satisfaction Problems. Volume 7 of SIAM Monographs on Discrete Mathematics and Applications. SIAM (2001)
12. Gallo, G., Simeone, B.: On the supermodular knapsack problem. Mathematical Programming **45** (1988) 295–309
13. Goldberg, A., Tarjan, R.: A new approach to the maximum flow problem. Journal of the ACM **35** (1988) 921–940
14. Iwata, S.: Submodular function minimization. Math. Progr. **112** (2008) 45–64
15. Jeavons, P., Cohen, D., Cooper, M.: Constraints, consistency and closure. Artificial Intelligence **101**(1–2) (1998) 251–265
16. Kolmogorov, V., Zabih, R.: What energy functions can be minimized via graph cuts? IEEE Trans. Pattern Anal. Mach. Intell. **26**(2) (2004) 147–159
17. Montanari, U.: Networks of constraints: Fundamental properties and applications to picture processing. Information Sciences **7** (1974) 95–132
18. Nemhauser, G., Wolsey, L.: Integer and Combinatorial Optimization. (1988)
19. Orlin, J.B.: A faster strongly polynomial time algorithm for submodular function minimization. In: IPCO'07. Volume 4513 of LNCS. (2007) 240–251
20. Rhys, J.: A selection problem of shared fixed costs and network flows. Management Science **17**(3) (1970) 200–207
21. Rossi, F., van Beek, P., Walsh, T., eds.: The Handbook of CP. Elsevier (2006)
22. Schiex, T., Fargier, H., Verfaillie, G.: Valued constraint satisfaction problems: hard and easy problems. In: IJCAI'95. (1995) 631–639
23. Schrijver, A.: A combinatorial algorithm minimizing submodular functions in strongly polynomial time. J. of Combinatorial Theory, Series B **80** (2000) 346–355
24. Schrijver, A.: Combinatorial Optimization: Polyhedra and Efficiency. Volume 24 of Algorithms and Combinatorics. Springer (2003)
25. Zalesky, B.: Efficient determination of Gibbs estimators with submodular energy functions. arXiv:math/0304041v1 (February 2008)