

1
2
3
4
5
6
7
8
9
10
11
12
13 Minimization of locally-defined submodular
14 functions by optimal soft arc consistency
15
16

17
18 Martin C. Cooper,
19 IRIT, University of Toulouse III, 31062 Toulouse, France
20 cooper@irit.fr
21
22

23
24 **Abstract**

25
26 Submodular function minimization is a polynomially-solvable com-
27 binatorial problem. Unfortunately the best known general-purpose
28 algorithms have high-order polynomial time complexity. In many ap-
29 plications the objective function is locally-defined in that it is the sum
30 of cost functions (also known as soft or valued constraints) whose ar-
31 ities are bounded by a constant. We prove that every Valued Constraint
32 Satisfaction Problem with submodular cost functions has an equivalent
33 instance on the same constraint scopes in which the actual minimum
34 value of the objective function is rendered explicit. Such an equiva-
35 lent instance is the result of establishing optimal soft arc consistency
36 and can hence be found by solving a linear program. From a practical
37 point of view, this provides us with an alternative algorithm for mini-
38 mizing locally-defined submodular functions. From a theoretical point
39 of view, this brings to light a previously-unknown connection between
40 submodularity and soft arc consistency.
41

42 **Keywords:** discrete optimization, valued constraint satisfaction prob-
43 lem, soft constraints, submodularity, majority operation, optimal soft arc
44 consistency, linear programming.
45

46
47 **Running head:** Minimization of submodular functions
48
49
50
51
52
53
54
55
56
57
58

1 Introduction

Soft (or valued) constraint satisfaction provides a framework which generalizes the Constraint Satisfaction Problem (CSP) [22] to allow for an optimization criterion [47, 42]. This criterion is the sum of cost functions, known as soft (or valued) constraints. Hard constraints (also known as crisp constraints) can naturally be coded in the same framework by cost functions which take on infinite values. This has allowed us to study, in the unifying framework of a generic constrained optimization problem over finite domains, such questions as

- what properties of the cost functions guarantee tractability [11, 12],
- which local reduction operations can simplify an instance, for example, by rendering explicit a good lower bound to be used in branch and bound search [17, 39, 16],
- what is the effect on algorithms of different properties of the set of possible costs [14, 15],

Applications of soft constraint satisfaction abound in Artificial Intelligence, Computer Vision and Bioinformatics [42].

Over boolean domains, there are only eight properties of cost functions which guarantee tractability [8, 11]. Of these, three can be considered as trivial and four can be considered as trivial extensions of tractable classes of crisp constraints, such as Horn clauses or 2SAT clauses. The remaining tractable class is the set of submodular soft constraints. Over non-boolean domains, the identification of all tractable classes of soft constraints remains an open problem. In the special case of $\{0, 1\}$ -valued soft constraints, we have conjectured that submodularity is again essentially the only reason for tractability [10]. Indeed, over size-3 domains, this has been shown to be the case [36].

Submodular function minimization (SFM) has numerous applications in Operations Research [24, 44, 51]. For example, the cut function of a graph [20] or a hypergraph is submodular [26]. Although it has been known for a long time that the ellipsoid algorithm can be used to solve SFM in polynomial time [28], this algorithm is not efficient in practice. Relatively recently, several new strongly polynomial combinatorial algorithms have been discovered for SFM [50, 31, 29, 30, 45]. These algorithms have been reviewed in detail by McCormick [41]. Unfortunately, the time complexity of the fastest published algorithms for SFM is $O((n^{4\gamma+n^5}) \min\{\log \hat{M}, n\})$, where n is the

total number of variables, γ is the time to evaluate the objective function f and \hat{M} is the maximum absolute value taken on by f [30, 45]. Nevertheless, for certain special cases of SFM, more efficient algorithms exist. For example, MINIMUM WEIGHTED CUT is a special case of SFM that can be solved in cubic time [27].

Given a Valued Constraint Satisfaction Problem (VCSP) \mathcal{P} , testing the existence of a zero-cost solution can be expressed as a CSP, which we denote by $\text{Bool}(\mathcal{P})$. We therefore briefly study CSPs in Section 2. In Section 3 we show that if \mathcal{P} has submodular cost functions of arbitrary arity, then $\text{Bool}(\mathcal{P})$ is nevertheless equivalent to a VCSP \mathcal{Q} with crisp *binary* submodular constraints. In Section 4 we show that soft arc consistency operations [17] preserve submodularity. It is known that a VCSP \mathcal{Q} with binary submodular constraints can be solved by finding a minimum cut (or equivalently, a maximum flow) in a directed graph $G_{\mathcal{Q}}$ [9]. We show, in Section 5, that for any non-zero flow in $G_{\text{Bool}(\mathcal{P})}$, there is a corresponding set of soft arc consistency operations which increases the value of the nullary constraint ϕ_0 in \mathcal{P} . In Section 6, we point out that after establishing optimal arc consistency [16] in \mathcal{P} , by definition, no such set of operations can increase ϕ_0 . This implies that the value of ϕ_0 after establishing optimal soft arc consistency is the cost of an optimal solution to \mathcal{P} .

We emphasize that since the complexity of our algorithm is exponential in the maximum arity of cost functions, it is only practical for the minimization of the sum of *locally-defined* submodular functions (i.e. objective functions which are the sum of cost functions of arity bounded by a constant k). This is an essential assumption that we make throughout the paper.

2 Constraint satisfaction and polymorphisms

In this section we present some terminology and notation used to describe the standard constraint satisfaction problem (CSP). The more general Valued Constraint Satisfaction Problem is defined in the next section.

Definition 2.1 *An instance of the **constraint satisfaction problem**, CSP, is a tuple $\mathcal{P} = \langle V, D, C \rangle$ where:*

- V is a finite set of n **variables**;
- $D = \{1, 2, \dots, d\}$ is a finite set of possible **values**;
- C is a set of **constraints**. Each element of C is a pair $c = \langle \sigma, R \rangle$ where σ is a tuple of variables called the **scope** of c , and R is a relation

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65

over D of arity $|\sigma|$ called the **constraint relation** of c .

Definition 2.2 For any CSP instance $\mathcal{P} = \langle V, D, C \rangle$, an **assignment** for \mathcal{P} is a mapping $s : V \rightarrow D$.

A **solution** to \mathcal{P} is an assignment which satisfies all of the constraints. That is, for each $\langle \sigma, R \rangle \in C$, the tuple $\langle s(v_1), s(v_2), \dots, s(v_r) \rangle \in R$, where $\sigma = \langle v_1, v_2, \dots, v_r \rangle$.

Example 2.3 The standard problem of colouring the vertices of a graph G with k colours so that adjacent vertices are assigned different colours can be viewed as a special case of the CSP, where the constraint relation of each constraint is the binary disequality relation, R_{\neq} , given by

$$R_{\neq} = \{ \langle a, b \rangle \in D^2 \mid a \neq b \}.$$

For any given graph $\langle V, E \rangle$, we have the corresponding CSP instance $\langle V, D, C \rangle$, where $D = \{1, 2, \dots, k\}$ and $C = \{ \langle \{v_i, v_j\}, R_{\neq} \rangle \mid \{v_i, v_j\} \in E \}$.

This problem is well-known to be NP-complete when $k \geq 3$. \square

If Γ is a set of relations over some fixed set D , we will write $\text{CSP}(\Gamma)$ to denote the class of all CSP instances where the constraint relations of all constraints lie in Γ .

For certain sets of relations Γ , the problem $\text{CSP}(\Gamma)$ is NP-complete. (For example, the set $\{R_{\neq}\}$, where R_{\neq} is the disequality relation over some set D with $|D| \geq 3$, as defined in Example 2.3.) For other sets of relations Γ , the problem $\text{CSP}(\Gamma)$ can be solved in polynomial time.

A finite set of relations Γ is called **tractable** if there exists a polynomial-time algorithm to solve $\text{CSP}(\Gamma)$. An infinite set of relations Γ is called tractable if all finite subsets of Γ are tractable. Many new tractable sets of relations have been identified by investigating certain invariance properties of relations, known as polymorphisms [7, 23, 34].

Definition 2.4 A function $f : D^m \rightarrow D$ is a **polymorphism** of a relation $R \subseteq D^r$ if for all $\langle a_{11}, \dots, a_{1r} \rangle, \dots, \langle a_{m1}, \dots, a_{mr} \rangle \in R$, we also have

$$\langle f(a_{11}, \dots, a_{m1}), \dots, f(a_{1r}, \dots, a_{mr}) \rangle \in R.$$

If a relation R has a polymorphism f , then we will say that R is **preserved by f** or that R is **f -closed**.

Example 2.5 A literal is a propositional variable X_i (known as a positive literal) or its negation $\neg X_i$ (known as a negative literal). A Horn clause is

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65

a disjunction of literals, at most one of which is positive. The relations over the 2-element domain $\{0, 1\}$ which are logically equivalent to a conjunction of Horn clauses are precisely the relations having the polymorphism $\min : \{0, 1\}^2 \rightarrow \{0, 1\}$, which returns the minimum of its 2 arguments [35].

A binary operation, \min , which returns the minimum of its two arguments, can be defined on any finite totally-ordered set D of arbitrary size. Hence, for each such D there is an obvious generalization to the set, Γ_{\min} , consisting of all min-closed relations over D . It has been shown [35] that $\text{CSP}(\Gamma_{\min})$ is tractable for all finite sets D . \square

Many other tractable sets of relations have been identified, or extended, thanks to the study of polymorphisms [4, 5, 6, 7, 34]. In fact, it is known that the existence of a non-trivial polymorphism of a set of relations Γ is a necessary condition for tractability of $\text{CSP}(\Gamma)$ [32, 23].

Definition 2.6 *A **majority operation** is a function $f : D^3 \rightarrow D$ satisfying*

$$\forall x, y \in D, \quad f(x, x, y) = f(x, y, x) = f(y, x, x) = x$$

A classic example of a majority operation is the function which returns the median of its three arguments, since $\text{median}(x, x, y) = x$. It has been shown that having a majority operation as a polymorphism is a sufficient condition for tractability of a set of relations [23, 34, 33]. However, it is the following more specific property of relations preserved by a majority operation which is of more interest to us in this paper.

Definition 2.7 *The **projection** of a relation R of arity r onto a pair of positions i and j , which we denote by $\text{pr}_{ij}R$, is the binary relation containing all pairs that can be extended to elements of R . That is,*

$$\text{pr}_{ij}R \stackrel{\text{def}}{=} \{\langle x_i, x_j \rangle \mid \langle x_1, \dots, x_r \rangle \in R\}.$$

*A relation R of arity r is said to be **decomposable into its binary projections** if*

$$R = \{\langle x_1, \dots, x_r \rangle \in D^r \mid \forall i, j \in \{1, \dots, r\}, \langle x_i, x_j \rangle \in \text{pr}_{ij}R\}.$$

Lemma 2.8 ([33]) *Any relation which is preserved by a majority operation is decomposable into its binary projections.*

When a relation is preserved by a majority operation, and hence decomposable into binary projections, this provides a very compact representation for the relation, by simply listing the binary projections.

3 Valued Constraint Satisfaction

In the constraint satisfaction problem, the aim is simply to find an assignment to the variables which satisfies all of the constraints. In other words, constraint satisfaction problems deal with *feasibility* rather than *optimization*. To provide a more general framework, the notion of an all-or-nothing constraint relation can be extended to the notion of a **cost function** which assigns a specified cost to each possible assignment. We use $\overline{\mathbb{R}}_+$ to denote $\{u \in \mathbb{R} : u \geq 0\} \cup \{\infty\}$.

Definition 3.1 For any set D , an order- r **cost function** on D is a function $\phi : D^r \rightarrow \overline{\mathbb{R}}_+$ which assigns a **cost** $\phi(a_1, \dots, a_r)$ to each combination of values $\langle a_1, \dots, a_r \rangle \in D^r$.

Definition 3.2 A cost function ϕ is said to be **crisp** if $\phi(x_1, \dots, x_r) \in \{0, \infty\}$ for all choices of $\langle x_1, \dots, x_r \rangle$.

A constraint relation can be modeled by a crisp cost function which assigns a cost of 0 to permitted assignments and a cost of ∞ to disallowed assignments.

Definition 3.3 An instance of the **valued constraint satisfaction problem**, VCSP, is a tuple $\mathcal{P} = \langle V, D, C \rangle$ where:

- V is a finite set of **variables**;
- $D = \{1, 2, \dots, d\}$ is a finite set of possible **values**;
- C is a set of **valued constraints**. Each element of C is a pair $c = \langle \sigma, \phi \rangle$ where σ is a tuple of variables called the **scope** of c , and ϕ is a mapping from $D^{|\sigma|}$ to $\overline{\mathbb{R}}_+$, called the **cost function** of c .

In the original and more general definition of the Valued Constraint Satisfaction Problem [47], costs were allowed to lie in any positive tomonoid: a totally ordered set S , with minimum element \perp and maximum element \top , closed under a commutative, associative, monotonic operation \oplus having \perp as identity element. Under the additional assumptions of discreteness and the existence of a partial inverse operation, it has been shown [15] that such a structure S can be decomposed into independent positive tomonoids, each of which is isomorphic to a subset of the natural numbers with the operator being either standard addition, $+$, or bounded addition, $+_m$, where $a +_m b = \min\{m, a + b\}$. The latter case is of some interest, because it can

be used to model the process of branch and bound search (m being the cost of the best solution found so far) [39]. However, for the purposes of this paper we shall restrict attention to the standard case studied in Mathematical Programming where all costs lie in $\overline{\mathbb{R}}_+$ and are combined using standard addition.

Definition 3.4 For any VCSP instance $\mathcal{P} = \langle V, D, C \rangle$, an **assignment** for \mathcal{P} is a mapping $s : V \rightarrow D$. The **cost** of an assignment s , denoted $Cost_{\mathcal{P}}(s)$, is given by the sum of the costs for the restrictions of s onto each constraint scope, that is,

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

An **optimal solution** to \mathcal{P} is an assignment with minimal cost.

Example 3.5 We can code the search for a minimum cut in a directed weighted graph G as a VCSP instance with a variable X_i for each node i of G and a valued constraint $\langle \langle X_i, X_j \rangle, \chi^{w_{ij}} \rangle$ for each directed edge $\langle i, j \rangle$ of weight $w_{ij} \in \overline{\mathbb{R}}_+$ in G , where

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

There are also two unary constraints $\langle \langle X_s \rangle, \mu_0 \rangle$ and $\langle \langle X_t \rangle, \mu_1 \rangle$, where s, t are the source and terminal nodes, respectively, and

$$\mu_u(x) = \begin{cases} \infty & \text{if } x \neq u \\ 0 & \text{if } x = u \end{cases}$$

The domain of all variables is $\{0, 1\}$. The minimum cut corresponds to the set of directed edges $\langle i, j \rangle$ such that $X_i = 0$ and $X_j = 1$. \square

In order to analyse the tractability of valued constraint satisfaction problems, we defined a generalization of the notion of polymorphism which is known as a **multimorphism** [11].

Definition 3.6 A list of functions, $\langle f_1, \dots, f_m \rangle$, where each f_i is a function from D^m to D , is a **multimorphism** of a cost function $\phi : D^r \rightarrow \overline{\mathbb{R}}_+$ if, for all $\langle a_{11}, \dots, a_{1r} \rangle, \dots, \langle a_{m1}, \dots, a_{mr} \rangle \in D^r$, we have

$$\sum_{i=1}^m \phi(f_i(a_{i1}, \dots, a_{i1}), \dots, f_i(a_{i1}, \dots, a_{i1})) \leq \sum_{i=1}^m \phi(a_{i1}, \dots, a_{i1})$$

If $\langle f_1, \dots, f_m \rangle$ is a multimorphism of a cost function ϕ , then the average cost of a set of m assignments is lowered by applying the functions f_1, \dots, f_m co-ordinatewise.

Example 3.7 A cost function ϕ has the multimorphism $\langle \min, \max \rangle$ if and only if ϕ satisfies the **submodularity** condition

$$\forall \bar{x}, \bar{y} \in D^r \quad \phi(\bar{x} \vee \bar{y}) + \phi(\bar{x} \wedge \bar{y}) \leq \phi(\bar{x}) + \phi(\bar{y})$$

where \vee and \wedge represent co-ordinatewise maximum and minimum operations respectively, i.e. $\bar{x} \vee \bar{y} = \langle \max(x[1], y[1]), \dots, \max(x[r], y[r]) \rangle$ where $\bar{x} = \langle x[1], \dots, x[r] \rangle$ and $\bar{y} = \langle y[1], \dots, y[r] \rangle$. \square

It is easy to verify that all unary cost functions (including μ_u) are submodular and that the cost function $\chi^w(x, y)$ of Example 3.5 is submodular. Many other examples of binary submodular functions can be found in [9]. Over the domain $D = \{1, 2, \dots, d\}$, examples of locally-defined submodular functions ϕ of arity $k > 2$ include:

- generalized 2-monotone functions [10], i.e. functions $\phi : D^r \rightarrow \{0, 1\}$ of the form

$$\phi(x_1, \dots, x_r) = \begin{cases} 0 & \text{if } (x_{i_1} \leq a_1 \wedge \dots \wedge x_{i_p} \leq a_p) \\ & \vee (x_{j_1} \geq b_1 \wedge \dots \wedge x_{j_q} \geq b_q) \\ \rho & \text{otherwise} \end{cases}$$

for some $a_1, \dots, a_p, b_1, \dots, b_q \in D$, $\rho \in \overline{\mathbb{R}}_+$ and $1 \leq i_1, \dots, i_p, j_1, \dots, j_q \leq r$, where p or q may be zero.

- The Euclidean distance function between two points $(x_1, x_2), (x_3, x_4)$ in the plane:

$$\phi(x_1, x_2, x_3, x_4) = \sqrt{(x_1 - x_3)^2 + (x_2 - x_4)^2}$$

We assume without loss of generality that each VCSP instance \mathcal{P} has a unary constraint $\langle \langle X_i \rangle, \phi_i \rangle$ for each variable as well as a nullary constraint $\langle \langle \rangle, \phi_0 \rangle$. This latter constraint has an empty scope and hence simply represents a constant contribution to the cost of each $\bar{x} \in D^n$. A nullary constraint is useful for storing a lower bound on $\text{Cost}_{\mathcal{P}}$.

For any cost function ϕ we define the corresponding sets of feasible and zero assignments in the following way.

Definition 3.8 For any cost function $\phi : D^r \rightarrow \overline{\mathbb{R}}_+$, the set of **feasible assignments** for ϕ , denoted $\text{Feas}(\phi)$, is defined as follows

$$\text{Feas}(\phi) \stackrel{\text{def}}{=} \{\bar{x} \in D^r : \phi(\bar{x}) < \infty\}$$

and the set of **zero assignments** for ϕ , denoted $\text{Bool}(\phi)$ is defined as follows

$$\text{Bool}(\phi) \stackrel{\text{def}}{=} \{\bar{x} \in D^r : \phi(\bar{x}) = 0\}$$

Lemma 3.9 [11, 13] If $\phi : D^r \rightarrow \overline{\mathbb{R}}_+$ has the multimorphism $\langle f_1, \dots, f_m \rangle$, then the relations $\text{Feas}(\phi)$ and $\text{Bool}(\phi)$ have the polymorphism f_i , for $i = 1, 2, \dots, m$.

Lemma 3.10 If ϕ is submodular, then $\text{Feas}(\phi)$ and $\text{Bool}(\phi)$ are decomposable into their binary projections.

Proof: If ϕ is submodular, then it follows from Lemma 3.9 that $\text{Feas}(\phi)$ and $\text{Bool}(\phi)$ are closed under the polymorphisms \max and \min . The operation median can be obtained from \max and \min by composition:

$$\text{median}(x, y, z) = \max(\min(x, y), \max(\min(y, z), \min(z, x)))$$

Hence, $\text{Feas}(\phi)$ and $\text{Bool}(\phi)$ are closed under the operation median , since the set of polymorphisms of a relation are closed under composition [32]. Therefore $\text{Feas}(\phi)$ and $\text{Bool}(\phi)$ are decomposable into their binary projections by Lemma 2.8, since median is a majority operation. ■

4 Submodularity-preserving transformations

Notation: If σ is a subset of the variables $V = \{X_1, \dots, X_n\}$, then we use D^σ to represent the set of possible assignments to the variables of σ . If $\bar{x} \in D^\sigma$ and $X_i \in \sigma$, then $\bar{x}[i]$ represents the value assigned to X_i by \bar{x} .

Definition 4.1 Two VCSPs $\mathcal{P}_1 = \langle V, D, C_1 \rangle$, $\mathcal{P}_2 = \langle V, D, C_2 \rangle$ are **equivalent** if $\forall t \in D^V$, $\text{Cost}_{\mathcal{P}_1}(t) = \text{Cost}_{\mathcal{P}_2}(t)$.

Definition 4.2 The **subproblem** of a VCSP $\langle V, D, C \rangle$ on $U \subseteq V$ is the problem VCSP(U) = $\langle U, D, C_U \rangle$, where $C_U = \{(\sigma, \phi) \in C : \sigma \subseteq U\}$.

```

1
2
3
4
5
6
7
8
9
10 Project( $\langle \sigma, \phi \rangle, i, a, \rho$ ):
11    $\phi_i(a) := \phi_i(a) + \rho$ ;
12   for all  $\bar{x} \in D^\sigma$  such that  $\bar{x}[i] = a$  do
13      $\phi(\bar{x}) := \phi(\bar{x}) - \rho$ ;
14
15 UnaryProject( $i, \rho$ ):
16   for all  $u \in D$  do
17      $\phi_i(u) := \phi_i(u) - \rho$ ;
18    $\phi_0 := \phi_0 + \rho$ ;
19
20
21 Shift( $\langle \sigma, \phi \rangle, i, j, a, b, \rho$ ):
22   for all  $p \in [1, a]$  do Project( $\langle \sigma, \phi \rangle, i, p, -\rho$ )
23   for all  $q \in [1, b]$  do Project( $\langle \sigma, \phi \rangle, j, q, \rho$ )
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65

```

Figure 1: Three submodularity-preserving EPTs.

Definition 4.3 For a VCSP $\langle V, D, C \rangle$, an *equivalence preserving transformation* (EPT) on $U \subseteq V$ is an operation which transforms the subproblem VCSP(U) into an equivalent VCSP.

In a VCSP, all costs are non-negative. This property must be preserved by an EPT so that ϕ_0 remains a lower bound on $\text{Cost}_{\mathcal{P}}$. Equivalence-preserving transformations which increase the value of ϕ_0 have proved very effective in speeding up the exhaustive search for an optimal solution to VCSPs in diverse applications [42]. They are mostly based on two basic operations: **Project** and **UnaryProject** [16]. For a given valued constraint $\langle \sigma, \phi \rangle$ and a given assignment $X_i = a$ (where $X_i \in \sigma$), **Project** adds a cost of ρ to $\phi_i(a)$, this increase being compensated by the subtraction of ρ from each $\phi(\bar{x})$ such that $\bar{x}[i] = a$. For a given variable X_i , **UnaryProject** adds a cost of ρ to the lower bound ϕ_0 , this increase being compensated by the subtraction of ρ from $\phi_i(u)$ for each $u \in D$. These operations are given in Figure 1.

The cost ρ may be positive or negative. We also allow the possibility that the cost ρ is infinite (i.e. $\rho \in \{-\infty, +\infty\}$). We extend the subtraction of real numbers to $\overline{\mathbb{R}}_+$ by defining $\infty - \alpha = \infty$ for all $\alpha \in \overline{\mathbb{R}}_+$. In particular, $\infty - \infty = \infty$.

Definition 4.4 For $S \subseteq \mathbb{R} \cup \{\infty\}$, we say that an operation is *S-closed* if, when applied to cost functions taking values in S , it returns cost functions

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65

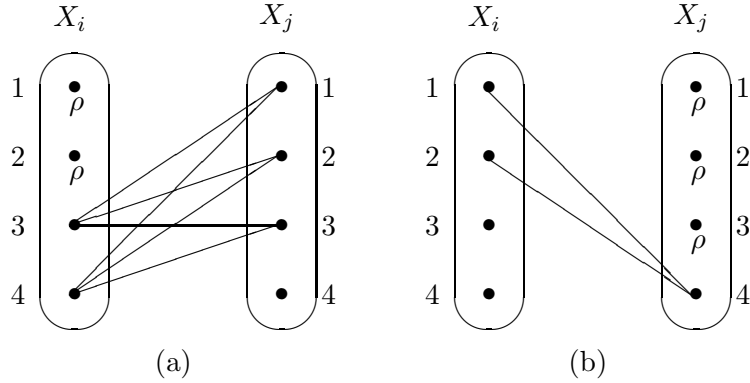


Figure 2: The two VCSPs (a) and (b) are equivalent.

whose values are also in S .

Thus, for example, $\mathbf{Project}(\langle \sigma, \phi \rangle, i, a, \rho)$ is $\overline{\mathbb{R}}_+$ -closed iff $\phi_i(a) + \rho \geq 0$ and for all $\bar{x} \in D^\sigma$ such that $\bar{x}[i] = a$, $\phi(\bar{x}) - \rho \geq 0$. We use the term **soft arc consistency (SAC) operation** to denote either an $\overline{\mathbb{R}}_+$ -closed **Project** operation or an $\overline{\mathbb{R}}_+$ -closed **UnaryProject** operation. The term soft arc consistency operation is used since these operations can be seen as generalizations of arc consistency operations in CSPs [17]. It is trivial to show that soft arc consistency operations are EPTs. It is also simple to verify that soft arc consistency operations preserve submodularity.

Figure 1 also gives the operation **Shift**. Whenever **Shift** is $\overline{\mathbb{R}}_+$ -closed, it is clearly also a submodularity-preserving EPT since it is just a sequence of **Project** operations. As an example of the operation **Shift**, consider the 2-variable VCSP shown in Figure 2(a). An oval represents a variable, the four small black circles inside each oval represent the values in the domain $D = \{1, 2, 3, 4\}$, a value ρ next to a domain value a in the domain of X_i represents a cost $\phi_i(a) = \rho$, and a line joining a in the domain of X_i with b in the domain of X_j represents a cost $\phi_{ij}(a, b) = \rho$. If $\rho < \infty$, then applying $\mathbf{Shift}(\langle \langle X_i, X_j \rangle, \phi_{ij} \rangle, i, j, 2, 3, \rho)$ to this VCSP produces the VCSP shown in Figure 2(b). For a non-negative cost $\rho \in \overline{\mathbb{R}}_+$, **Shift** is $\overline{\mathbb{R}}_+$ -closed iff $\forall p \in [1, a], \phi_i(p) \geq \rho$ and $\forall p \in [a + 1, d], \forall q \in [1, b], \forall \bar{x} \in D^\sigma, \bar{x}[i] = p \wedge \bar{x}[j] = q \Rightarrow \phi(\bar{x}) \geq \rho$.

In order to give an alternative definition of **Shift**, we need the following definition.

Definition 4.5 [9] Let $D = \{1, \dots, d\}$. A binary function $\phi : D^2 \rightarrow \overline{\mathbb{R}}_+$ is

1
2
3
4
5
6
7
8
9
10 *a generalized interval constraint on D if it has the following form:*

$$11 \quad \phi(x, y) = \begin{cases} 0 & \text{if } (x < a) \vee (y > b) \\ \rho & \text{otherwise} \end{cases}$$

12
13
14 for some $a, b \in \{1, \dots, d+1\}$ and some $\rho \in \overline{\mathbb{R}}_+$. Such a function is denoted
15 $\eta_{[a,b]}^\rho$.

16
17 We can explain the name of these functions by the fact that the unary func-
18 tion $g(x) = \eta_{[a,b]}^\rho(x, x)$ returns the value ρ if and only if its argument lies in
19 the interval $[a, b]$. Outside of this interval g returns the value 0. Generalized
20 interval constraints are easily shown to be submodular [9]. In the VCSP
21 of Figure 2(a), $\phi_i(X_i)$ is $\eta_{[1,2]}^\rho(X_i, X_i)$ and $\phi_{ij}(X_i, X_j)$ is $\eta_{[3,3]}^\rho(X_i, X_j)$. The
22 equivalence between the two VCSPs in Figure 2 is just one example of the
23 following more general result.
24
25

26
27 **Lemma 4.6** For all $a, b \in D = \{1, 2, \dots, d\}$ and for all $\rho \in \overline{\mathbb{R}}_+$,

$$28 \quad \eta_{[1,a]}^\rho(x, x) + \eta_{[a+1,b]}^\rho(x, y) = \eta_{[1,b]}^\rho(y, y) + \eta_{[b+1,a]}^\rho(y, x)$$

29
30 **Proof:** It is easy to verify that both the left-hand side and the right-hand
31 side of the equality are equal to the function $\psi : D^2 \rightarrow \overline{\mathbb{R}}_+$ given by

$$32 \quad \psi(x, y) = \begin{cases} \rho & \text{if } (x \leq a) \vee (y \leq b) \\ 0 & \text{otherwise} \end{cases}$$

33
34
35
36
37
38 ■

39 **Shift** corresponds to subtracting the left-hand side of the equation in
40 Lemma 4.6 and adding the right-hand side of the same equation. We can
41 therefore rewrite **Shift** as shown in Figure 3.

42
43 In CSPs, local consistency operations [22] preserve polymorphisms. This
44 is an important property which has led to the identification of many classes
45 of tractable crisp constraints for which different levels of local consistency
46 provide a solution procedure [33, 6]. In VCSPs, soft local consistency oper-
47 ations [15] do not, in general, preserve multimorphisms (and, in particular,
48 submodularity). As an example, consider a VCSP with a ternary cost func-
49 tion $\langle\langle X_1, X_2, X_3 \rangle, \phi_{123} \rangle$, and a binary cost function $\langle\langle X_1, X_2 \rangle, \phi_{12} \rangle$, where
50 ϕ_{123}, ϕ_{12} are both identically equal to 1, (and hence trivially submodular).
51 If a weight of 1 is projected from ϕ_{123} onto $\phi_{12}(0, 0)$ then function ϕ_{12} is no
52 longer submodular. On the other hand, the submodularity-preserving EPTs
53 given in Figure 1 can be applied any number of times while guaranteeing
54 that the cost functions of the resulting equivalent problem are submodular.
55
56
57
58

```

9 Shift( $\langle \sigma, \phi \rangle, i, j, a, b, \rho$ ):
10   for all  $\bar{x} \in D^\sigma$  do
11      $\phi(\bar{x}) := \phi(\bar{x}) - \eta_{[a+1,b]}^\rho(\bar{x}[i], \bar{x}[j]) + \eta_{[b+1,a]}^\rho(\bar{x}[j], \bar{x}[i]);$ 
12   for all  $u \in D$  do
13      $\phi_i(u) := \phi_i(u) - \eta_{[1,a]}^\rho(u, u);$ 
14      $\phi_j(u) := \phi_j(u) + \eta_{[1,b]}^\rho(u, u)$ 
15   end for;

```

Figure 3: Another way of writing the EPT **Shift**.

5 A lower-bound-increasing path transformation

Let $c = \langle \sigma, \phi \rangle$ be a valued constraint. Consider the crisp constraint $\langle \sigma, \text{Bool}(\phi) \rangle$ where we recall that $\text{Bool}(\phi)$ is the relation defined by, for all $\bar{x} \in D^\sigma$,

$$\bar{x} \in \text{Bool}(\phi) \Leftrightarrow \phi(\bar{x}) = 0$$

For $X_i, X_j \in \sigma$, let $R_{ij}^c = \text{pr}_{ij} \text{Bool}(\phi)$ be the binary projection onto variables X_i, X_j of $\text{Bool}(\phi)$. If ϕ is submodular then, by Lemma 3.10, $\text{Bool}(\phi)$ is decomposable into its binary projections.

Consider a VCSP instance $\mathcal{P} = \langle V, D, C \rangle$ with submodular valued constraints. Suppose that $\phi_0 = 0$. Now, for $\bar{x} \in D^V$, $\text{Cost}_{\mathcal{P}}(\bar{x}) = 0$ if and only if \bar{x} satisfies the crisp constraint $\langle \sigma, \text{Bool}(\phi) \rangle$ for all valued constraints $c = \langle \sigma, \phi \rangle$ of \mathcal{P} . Hence, by Lemma 3.10, $\text{Cost}_{\mathcal{P}}(\bar{x}) = 0$ if and only if \bar{x} satisfies the binary crisp constraint $\langle \langle X_i, X_j \rangle, R_{ij}^c \rangle$ for all valued constraints $c = \langle \sigma, \phi \rangle$ of \mathcal{P} and for all $X_i, X_j \in \sigma$.

We denote by $\text{Bool}(\mathcal{P})$ the CSP whose constraints are exactly these binary crisp constraints, i.e. $\text{Bool}(\mathcal{P}) = \langle V, D, \text{Bool}(C) \rangle$ where $\text{Bool}(C) = \{ \langle \langle X_i, X_j \rangle, R_{ij}^c \rangle : c = \langle \sigma, \phi \rangle \in C, X_i, X_j \in \sigma \}$. Note that $\text{Bool}(\mathcal{P})$ contains unary constraints (since we can have $i = j$) but no nullary constraint. $\text{Bool}(\mathcal{P})$ can be solved in polynomial time since all its constraints are max-closed (and also min-closed) [35]. An alternative polynomial-time algorithm for $\text{Bool}(\mathcal{P})$ is obtained by observing that it is a special case (since it involves only zero or infinite costs) of the minimization of the sum f of binary submodular functions [9]. This latter algorithm constructs a graph $G_{\mathcal{P}}$ such that the maximum flow, and hence the minimum cut, of $G_{\mathcal{P}}$ has a value equal to the minimum value of f . An interesting property of $G_{\mathcal{P}}$ is that any strictly positive flow in $G_{\mathcal{P}}$ corresponds to a set of equivalence-preserving transformations in \mathcal{P} that preserve submodularity and produce a better lower bound for \mathcal{P} .

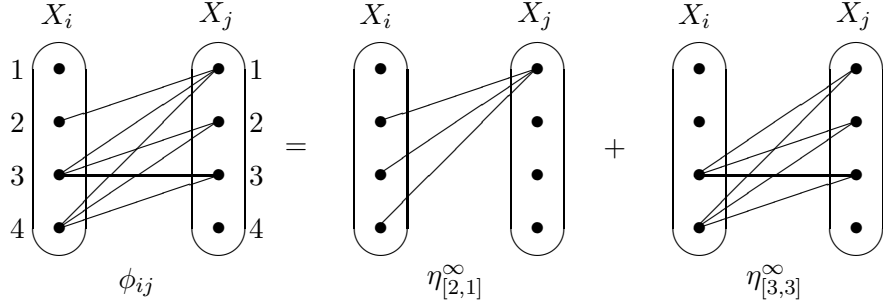


Figure 4: Example of the decomposition of a binary submodular function ϕ_{ij} into generalized interval constraints.

An obvious idea is to find and apply such EPTs until the final version of \mathcal{P} is such that $\text{Bool}(\mathcal{P})$ has a solution. The lower bound stored in ϕ_0 would then be the minimum value of $\text{Cost}_{\mathcal{P}}$. Unfortunately, since the increase in ϕ_0 may become smaller and smaller at each iteration, this process is not guaranteed to terminate in finite time. A solution to this problem will be presented in the following section.

It is known that any binary submodular function $\psi : D^2 \rightarrow \overline{\mathbb{R}}_+$ can be expressed as the sum of generalized interval constraints [9]. Although the binary constraints of $\text{Bool}(\mathcal{P})$ correspond to relations R_{ij}^c , we can clearly identify R_{ij}^c with the corresponding binary cost function ϕ_{ij} which returns a value of 0 if $(x, y) \in R_{ij}^c$ and a value of ∞ if $(x, y) \notin R_{ij}^c$. Figure 4 shows an example of the decomposition of a binary submodular function ϕ_{ij} . In this example, $\phi_{ij} = \eta_{[2,1]}^\infty + \eta_{[3,3]}^\infty$. In Figure 4, each line joining a in the domain of X_i and b in the domain of X_j represents a cost $\phi_{ij}(a, b) = \infty$.

To express ϕ_{ij} as the sum of generalized interval constraints, we can use the following procedure, which simply performs an exhaustive search for all $\eta_{[a,b]}^\infty$ which satisfy $\eta_{[a,b]}^\infty \leq \phi_{ij}$. Recall that $D = \{1, \dots, d\}$. Set

$$q = \max\{u \in D \cup \{0\} : 1 \leq y \leq u \Rightarrow \phi_{ij}(d, y) = \infty\}$$

In the example of Figure 4, $q = 3$. Let $t_0 = 1$ and then for $r = 1, \dots, q$, set

$$t_r = \min\{t \in [t_{r-1}, d] : d \geq x \geq t \Rightarrow \phi_{ij}(x, r) = \infty\}$$

In the example, we find $t_1 = 2$, $t_2 = t_3 = 3$, meaning that $\eta_{[2,1]}^\infty \leq \phi_{ij}$, $\eta_{[3,2]}^\infty \leq \phi_{ij}$, $\eta_{[3,3]}^\infty \leq \phi_{ij}$. Similarly, set

$$p = \max\{v \in D \cup \{0\} : 1 \leq x \leq v \Rightarrow \phi_{ij}(x, d) = \infty\}$$

In the example of Figure 4, $p = 0$. Let $s_0 = 1$ and then for $r = 1, \dots, p$, set

$$s_r = \min\{s \in [s_{r-1}, d] : d \geq y \geq s \Rightarrow \phi_{ij}(r, y) = \infty\}$$

Then ϕ_{ij} is the sum of the generalized interval constraints:

$$\begin{aligned} \eta_{[t_r, r]}^\infty(X_i, X_j) \quad (r = 1, \dots, q) \quad \text{and} \\ \eta_{[s_r, r]}^\infty(X_j, X_i) \quad (r = 1, \dots, p) \end{aligned}$$

Note that, in this decomposition, $\eta_{[t_r, r]}^\infty(X_i, X_j)$ is redundant if $t_r = t_{r+1}$ (for $1 \leq r < q$) and, similarly, $\eta_{[s_r, r]}^\infty(X_j, X_i)$ is redundant if $s_r = s_{r+1}$ (for $1 \leq r < p$). In the example of Figure 4, we therefore find exactly the decomposition shown in the figure. The decomposition of any crisp binary submodular function ϕ_{ij} involves $O(d)$ generalized interval constraints and can clearly be found in $O(d^2)$ time by the above procedure.

Definition 5.1 Let $\mathcal{P} = \langle V, \{1, \dots, d\}, C \rangle$ be an instance of VCSP. We denote by $GIC_{\mathcal{P}}$ the set of all the generalized interval constraints derived from the decompositions of the constraints of $\text{Bool}(\mathcal{P})$. We define the directed graph $G_{\text{Bool}(\mathcal{P})}$ as follows.

- The vertices of $G_{\text{Bool}(\mathcal{P})}$ are

$$\{s, t\} \cup \{x^u \mid x \in V, u \in \{0, 1, \dots, d\}\}$$

- The edges of $G_{\text{Bool}(\mathcal{P})}$ are as follows:

- for each $x \in V$, there is an edge from s to x^d ;
- for each $x \in V$, there is an edge from x^0 to t ;
- for each $x \in V$ and each $u \in \{1, 2, \dots, d-2\}$, there is an edge from x^u to x^{u+1} .
- for each generalized interval constraint $\langle \langle x, y \rangle, \eta_{[a, b]}^\infty \rangle \in GIC_{\mathcal{P}}$, there is an edge from y^b to x^{a-1} . These are called “constraint edges”.

Figure 5 shows a 3-variable VCSP \mathcal{P} with submodular cost functions. It consists of four generalized interval constraints: two unary constraints ($\eta_{[1, 3]}^1(x, x)$ and $\eta_{[3, 4]}^1(z, z)$) and two binary constraints ($\eta_{[3, 3]}^1(x, y)$ and $\eta_{[4, 2]}^1(y, z)$). The corresponding graph $G_{\text{Bool}(\mathcal{P})}$ is shown in Figure 6.

The graph $G_{\text{Bool}(\mathcal{P})}$ is an unweighted version of the graph defined in [9] (which was itself inspired by a similar graph given by Creignou et al. [18, 19]).

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65

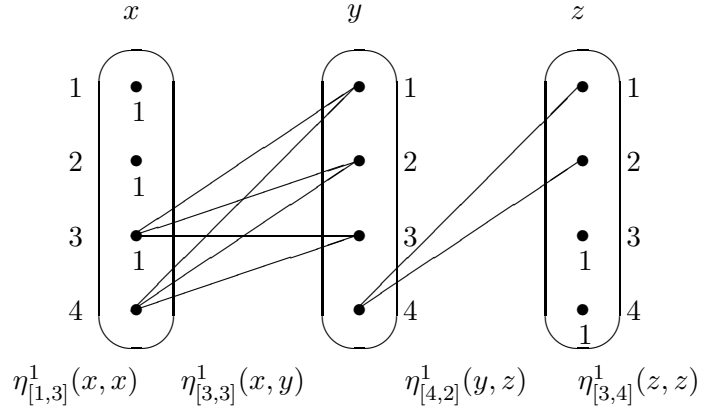


Figure 5: An example of a 3-variable VCSP composed of four generalized interval constraints.

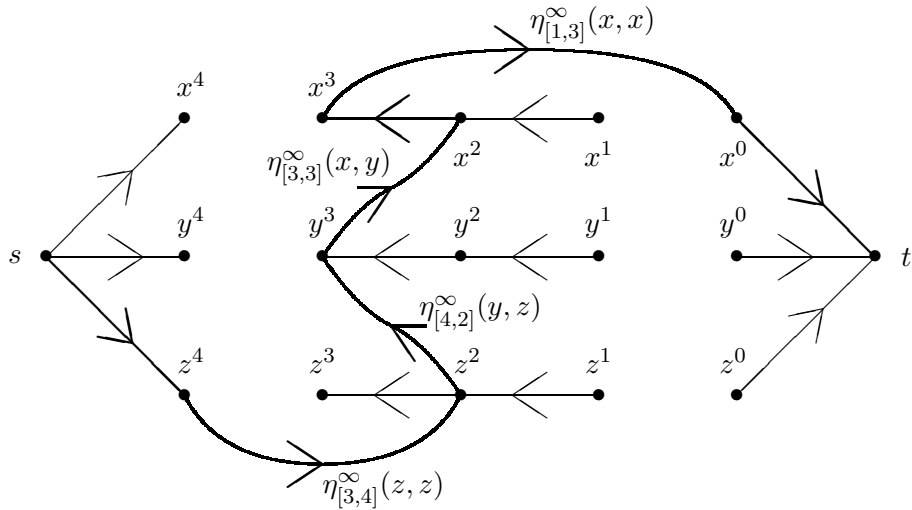


Figure 6: The graph $G_{\text{Bool}(\mathcal{P})}$ for the VCSP in Figure 5 with a path from the source s to the terminus t (shown in thick lines).

Lemma 5.2 *If there is no path from s to t in the directed graph $G_{\text{Bool}(\mathcal{P})}$, then $\text{Cost}_{\mathcal{P}}(\bar{x}) = 0$ where $\bar{x} \in D^n$ is given by*

$$\bar{x}[i] = \min\{u | X_i^u \in N_s\} \quad (1)$$

where N_s is the set of nodes of $G_{\text{Bool}(\mathcal{P})}$ connected to s .

If, on the other hand, there is a path π from s to t in $G_{\text{Bool}(\mathcal{P})}$, then the set of constraint edges on π , in inverse order, correspond to a set of generalized interval constraints of the following form: $\eta_{[1+a_j, b_j]}^\infty(X_{i_{j-1}}, X_{i_j})$ ($j = 1, \dots, p$), where

- $a_1 = 0$ and $b_p = d$;
- $\forall j = 1, \dots, p-1, b_j \geq a_{j+1}$.

Proof: Suppose that there is no path from s to t in $G_{\text{Bool}(\mathcal{P})}$, but that $\text{Cost}_{\mathcal{P}}(\bar{x}) > 0$ where $\bar{x} \in D^n$ is given by Equation (1). This means that \bar{x} is not a solution to $\text{Bool}(\mathcal{P})$ and hence that \bar{x} violates some generalized interval constraint $\langle \langle X_i, X_j \rangle, \eta_{[a, b]}^\infty \rangle \in \text{GIC}_{\mathcal{P}}$. Therefore, $\bar{x}[i] \geq a$ and $\bar{x}[j] \leq b$. By definition of N_s , there is a path in $G_{\text{Bool}(\mathcal{P})}$ from s to $X_j^{\bar{x}[j]}$. By construction of $G_{\text{Bool}(\mathcal{P})}$, this path can be extended to X_j^b (since $\bar{x}[j] \leq b$) and then to X_i^{a-1} (since $\langle \langle X_i, X_j \rangle, \eta_{[a, b]}^\infty \rangle \in \text{GIC}_{\mathcal{P}}$). Thus $X_i^{a-1} \in N_s$, which contradicts the minimality of $\bar{x}[i]$ since $a-1 < \bar{x}[i]$.

The second half of the Lemma follows directly from the construction of $G_{\text{Bool}(\mathcal{P})}$. ■

Let π be a path from s to t in $G_{\text{Bool}(\mathcal{P})}$. For each generalized interval constraint $\eta_{[1+a_j, b_j]}^\infty(X_{i_{j-1}}, X_{i_j})$ corresponding to a constraint edge on π , there is a constraint $c^j = \langle \sigma^j, \phi^j \rangle$ satisfying $\phi^j(\bar{x}) > 0$, for all $\bar{x} \in D^{|\sigma^j|}$ such that $(\bar{x}[i_{j-1}] \geq 1 + a_j) \wedge (\bar{x}[i_j] \leq b_j)$. The path from s to t in the graph $G_{\text{Bool}(\mathcal{P})}$ shown in Figure 6 corresponds, in inverse order, to the sequence of generalized interval constraints $\eta_{[1, 3]}^\infty(x, x)$, $\eta_{[3, 3]}^\infty(x, y)$, $\eta_{[4, 2]}^\infty(y, z)$, $\eta_{[3, 4]}^\infty(z, z)$ in $\text{Bool}(\mathcal{P})$.

Lemma 5.3 *Let π be a path from s to t in $G_{\text{Bool}(\mathcal{P})}$ and let c^1, \dots, c^p be the constraints corresponding to the constraint edges of π in the inverse order to that in which they are visited by π . Let $a_{p+1} = d$. Then there exists a real number $\rho > 0$ such that applying **Shift**($c^j, i_{j-1}, i_j, a_j, a_{j+1}, \rho$) for $j = 1, \dots, p$ (in this order), followed by **UnaryProject**(i_p, ρ), is an EPT which increases the lower bound ϕ_0 of \mathcal{P} by ρ .*

1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65

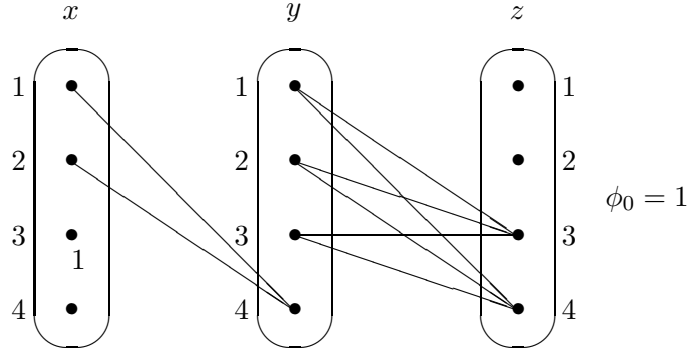


Figure 7: The result of applying a path EPT to the VCSP of Figure 5.

Proof: $\text{Shift}(c^j, i_{j-1}, i_j, a_j, a_{j+1}, \rho)$ subtracts $\eta_{[1+a_j, a_{j+1}]}^\rho(X_{i_{j-1}}, X_{i_j})$ from ϕ^j . Let μ be the minimum non-zero value taken on by the cost functions of \mathcal{P} . For $\rho > 0$ sufficiently small, for example $\rho = \mu/p$, this is possible (in the sense that the resulting cost function takes on non-negative values) due to the following three facts:

- $\phi^j(\bar{x}) > 0$, for all $\bar{x} \in D^{|\sigma^j|}$ such that $(\bar{x}[i_{j-1}] \geq 1 + a_j) \wedge (\bar{x}[i_j] \leq b_j)$,
- $\eta_{[1+a_j, a_{j+1}]}^\rho(x, y) = \rho$ only for $(x \geq 1 + a_j) \wedge (y \leq a_{j+1})$ (and is equal to 0 otherwise),
- $b_j \geq a_{j+1}$

$\text{Shift}(c^j, i_{j-1}, i_j, a_j, a_{j+1}, \rho)$ subtracts $\eta_{[1, a_j]}^\rho(X_{i_{j-1}}, X_{i_{j-1}})$ from $\phi_{i_{j-1}}$ and adds $\eta_{[1, a_{j+1}]}^\rho(X_{i_j}, X_{i_j})$ to ϕ_{i_j} . Now, $\eta_{[1, a_1]}^\rho(X_{i_0}, X_{i_0})$ is identically equal to zero, since $a_1 = 0$. For $j = 2, \dots, p$, it is possible to subtract $\eta_{[1, a_j]}^\rho(X_{i_{j-1}}, X_{i_{j-1}})$ from $\phi_{i_{j-1}}$ since this same generalized interval constraint was added to $\phi_{i_{j-1}}$ by the previous call to **Shift**, namely $\text{Shift}(c^{j-1}, i_{j-2}, i_{j-1}, a_{j-1}, a_j, \rho)$. The generalized interval constraint $\eta_{[1, a_{p+1}]}^\rho(X_{i_p}, X_{i_p})$, which is added to ϕ_{i_p} in the final call of **Shift**, is identically equal to ρ , since $a_{p+1} = d$. It is then possible to apply the operation **UnaryProject**(i_p, ρ), the final operation of the EPT, which subtracts ρ from ϕ_{i_p} and adds ρ to the nullary constraint ϕ_0 . ■

In order to give a simple illustration of the EPT corresponding to a path π in $G_{\text{Bool}(\mathcal{P})}$ (which we call a **path EPT**), consider the path shown in Figure 6. Figure 7 shows the result of applying the corresponding EPT. This path EPT is equivalent to subtracting $\eta_{[1,2]}^1(x, x) + \eta_{[3,3]}^1(x, y) + \eta_{[4,2]}^1(y, z)$

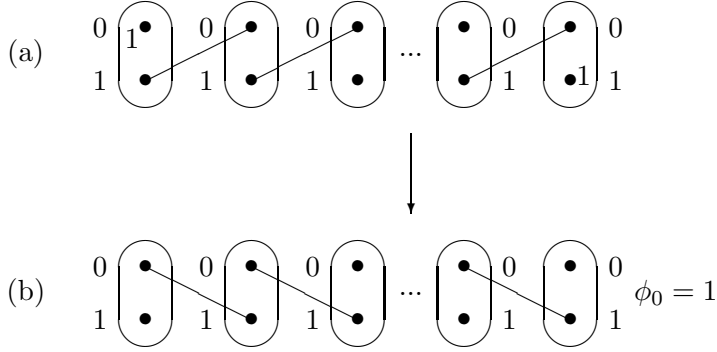


Figure 8: A path EPT corresponding to the equivalence between objective functions given in Equation 2.

+ $\eta_{[3,4]}^1(z, z)$ and adding $\eta_{[4,2]}^1(y, x) + \eta_{[3,3]}^1(z, y) + 1$. We can see that this is an EPT since, by two applications of Lemma 4.6,

$$\begin{aligned}
& \eta_{[1,2]}^1(x, x) + \eta_{[3,3]}^1(x, y) + \eta_{[4,2]}^1(y, z) + \eta_{[3,4]}^1(z, z) \\
&= \eta_{[4,2]}^1(y, x) + \eta_{[3,3]}^1(z, y) + \eta_{[1,2]}^1(z, z) + \eta_{[3,4]}^1(z, z) \\
&= \eta_{[4,2]}^1(y, x) + \eta_{[3,3]}^1(z, y) + 1
\end{aligned}$$

As another example, consider the simple case in which \mathcal{P} contains only unary and binary valued constraints over the boolean domain $D = \{0, 1\}$. Then the following well-known equivalence between objective functions [3]

$$x_1^c + x_1x_2^c + \dots + x_{p-2}x_{p-1}^c + x_p = 1 + x_1^c x_2 + \dots + x_{p-1}^c x_p \quad (2)$$

(where $x_i^c = 0 \Leftrightarrow x_i = 1$) corresponds to the path EPT with $\rho = 1$, illustrated by the transformation from the VCSP of Figure 8(a) to the VCSP of Figure 8(b).

Theorem 5.4 *Let \mathcal{P} be a VCSP with e submodular cost functions of arity bounded by a constant $k \geq 2$ over domains of size d . Determining whether $\min \text{Cost}_{\mathcal{P}}$ is 0 or infinite can both be achieved in $O(ed^k)$ time. If $0 < \min \text{Cost}_{\mathcal{P}} < \infty$, then \mathcal{P} can be transformed in $O(ed^{k+1})$ time via a path EPT into an equivalent VCSP with submodular cost functions and with an explicit lower bound $\phi_0 > 0$.*

Proof: First of all, notice that determining whether there exists some $\bar{x} \in D^n$ such that $\text{Cost}_{\mathcal{P}} < \infty$ (respectively, such that $\text{Cost}_{\mathcal{P}} = 0$) involves solving the CSP composed of the binary projections of the max-closed and

min-closed constraints $\langle \sigma, \text{Feas}(\phi) \rangle$ (respectively, $\langle \sigma, \text{Bool}(\phi) \rangle$) for each valued constraint $\langle \sigma, \phi \rangle$ of \mathcal{P} . This can be achieved in $O(ed^k)$ time for the calculation of the binary projections plus $O(ed^2)$ time to solve the resulting CSP by establishing arc consistency [35, 1].

In the following, we can therefore assume that $\min \text{Cost}_{\mathcal{P}}$ is strictly positive and finite. The algorithm to increase ϕ_0 involves three stages:

1. the construction of $G_{\text{Bool}(\mathcal{P})}$;
2. the search for a path π from s to t ;
3. applying the EPT corresponding to such a path π .

The construction of the graph $G_{\text{Bool}(\mathcal{P})}$ can be achieved in $O(ed^k)$ time; the calculation of the binary projections $\langle \sigma, \text{Bool}(\phi) \rangle$ requires $O(ed^k)$ time and the decomposition of the resulting binary constraints into the sum of generalized interval constraints requires $O(ed^2)$ time. The number of edges in $G_{\text{Bool}(\mathcal{P})}$ is $O(ed)$, since each of the (at most C_k^2) binary projections of each constraint is decomposed into $O(d)$ generalized interval constraints. Using appropriate data structures, finding a path from s to t (which exists, by Lemma 5.2, by our assumption that $\min \text{Cost}_{\mathcal{P}} > 0$) requires $O(ed)$ operations since this is an upper bound on the number of edges in $G_{\text{Bool}(\mathcal{P})}$. Applying the EPT corresponding to a path π has time complexity $O(ed^{k+1})$, since **Shift** has complexity $O(d^k)$ and the number of edges in a path π is bounded above by the total number of edges in $G_{\text{Bool}(\mathcal{P})}$. Thus, the time complexity to increase ϕ_0 by $\rho > 0$ is $O(ed^{k+1})$. As in the proof of Lemma 5.3, we can choose $\rho = \mu/p$ (where μ is the minimum non-zero value taken on by the cost functions of \mathcal{P}). Such a value of ρ can be calculated in $O(ed^k)$ time. ■

Consider a path π as in Lemmas 5.2 and 5.3. For each constraint $c = \langle \sigma, \phi \rangle$ of $\mathcal{P} = \langle V, D, C \rangle$ and for each $\bar{x} \in D^\sigma$, let

$$N(c, \bar{x}) = |\{j : 1 \leq j \leq p, c^j = c, (\bar{x}[i_{j-1}] \geq 1 + a_j) \wedge (\bar{x}[i_j] \leq a_{j+1})\}| \\ - |\{j : 1 \leq j \leq p, c^j = c, (\bar{x}[i_j] \geq 1 + a_{j+1}) \wedge (\bar{x}[i_{j-1}] \leq a_j)\}|$$

This is the number of times ρ will be subtracted from $\phi(\bar{x})$ when the path EPT corresponding to π is applied. For example, suppose that $\sigma = \langle X_1, X_2, X_3, X_4 \rangle$, $\bar{x} = \langle 2, 1, 2, 1 \rangle$ and π contains the edges $\langle X_1^1, X_2^1 \rangle$ and $\langle X_3^1, X_4^1 \rangle$, both derived from the constraint $\langle \sigma, \phi \rangle$. Applying the path EPT corresponding to π requires calculating

$$\phi - \eta_{[2,1]}^\rho(X_1, X_2) - \eta_{[2,1]}^\rho(X_3, X_4)$$

and hence we have to subtract 2ρ from $\phi(2, 1, 2, 1)$.

The maximum value of ρ which guarantees that costs remain non-negative is

$$\min\{\phi(\bar{x})/N(c, \bar{x}) : c = \langle \sigma, \phi \rangle \in C, \bar{x} \in D^\sigma, N(c, \bar{x}) > 0\} \quad (3)$$

Assume that the original costs in \mathcal{P} are all integers or infinite. Let M represent $\min \text{Cost}_{\mathcal{P}}$. If we could always choose an integer value for ρ , then after applying at most M path EPTs we would be guaranteed to find an equivalent submodular VCSP \mathcal{P}' with $\text{Bool}(\mathcal{P}')$ solvable. Unfortunately, the value of ρ given by Equation (3) may be less than 1.

We can nevertheless identify a case in which ρ can always be chosen to be an integer, namely the case of at most ternary cost functions over the boolean domain $D = \{1, 2\}$. By construction, any path π from s to t in $G_{\text{Bool}(\mathcal{P})}$ passes through just one node of the form x^d and just one node of the form y^0 , for some variables x, y . When $d = 2$, all other nodes on π must therefore be of the form z^1 , for variables z . Thus, π (in inverse order) corresponds to a sequence of generalized interval constraints of the form

$$\eta_{[1+a_j, b_j]}^\infty(X_{i_{j-1}}, X_{i_j}) \quad (j = 1, \dots, p)$$

where $a_1 = 0$, $b_p = 2$ and, for $j = 1, \dots, p-1$, $b_j = a_{j+1} = 1$, or in other words:

$$\eta_{[1,1]}^\infty(X_{i_0}, X_{i_1}), \eta_{[2,1]}^\infty(X_{i_1}, X_{i_2}), \dots, \eta_{[2,1]}^\infty(X_{i_{p-2}}, X_{i_{p-1}}), \eta_{[2,2]}^\infty(X_{i_{p-1}}, X_{i_p})$$

If we assume that \mathcal{P} is soft arc consistent [17], then we must have $i_0 = i_1$ (otherwise there is some constraint $\langle c, \phi \rangle$ bounded below by $\eta_{[1,1]}^1(X_{i_0}, X_{i_1})$ which would allow us to project a weight of 1 onto $\phi_{i_1}(1)$, thus contradicting soft arc consistency). Similarly, we can deduce that $i_{p-1} = i_p$ and hence that the constraint edges (X_{i_0}, X_{i_1}) and $(X_{i_{p-1}}, X_{i_p})$ correspond to unary constraints. Assuming, without loss of generality, that π is cycle-free, the variables X_{i_j} ($j = 1, \dots, p-1$) are distinct. It follows that any set σ of up to three variables (the scope of an at-most-ternary constraint c) can only include two of the scopes $\{X_{i_{j-1}}, X_{i_j}\}$, $\{X_{i_{j'-1}}, X_{i_{j'}}\}$ ($2 \leq j < j' \leq p-1$) if $i_j = i_{j'-1}$. But when $i_j = i_{j'-1}$, there is no $\bar{x} \in D^\sigma$ such that $\bar{x}[i_j] \leq a_{j+1} = 1$ and $\bar{x}[i_{j'-1}] = \bar{x}[i_j] \geq 1 + a_j = 2$. Therefore $N(c, \bar{x}) \leq 1$ for all $\bar{x} \in D^\sigma$, which guarantees the integrality of ρ . In this special case of at most ternary constraints over a boolean domain, a cubic-time algorithm is already known for SFM [2] by reduction to MINCUT. An interesting question which arises is whether there are other conditions which guarantee $N(c, \bar{x}) \leq 1$.

6 SFM via Linear Programming

We now consider a direct application of Theorem 5.4. We show in this section that, given an instance \mathcal{P} of locally-defined SFM, there is always an equivalent problem \mathcal{P}' on the same scopes in which the minimum value of the objective function has been rendered explicit. Furthermore, \mathcal{P}' can be found by establishing optimal soft arc consistency.

We have seen in the previous section that once an equivalent VCSP \mathcal{P}' has been found in which $\phi_0 = M$ (the minimum value of the submodular function $\text{Cost}_{\mathcal{P}}$), finding a minimizer is straightforward, since it is equivalent to solving $\text{Bool}(\mathcal{P}')$, a CSP with max-closed constraints [35]. We show in this section that such an equivalent problem \mathcal{P}' can be found by solving a linear program.

By definition, **Shift** is equivalent to a sequence of **Project** operations. Furthermore, it is easy to verify that in a path EPT all intermediate problems are valid VCSPs in that all costs are nonnegative. Hence, a path EPT is equivalent to a sequence of soft arc consistency operations. We now introduce a more general transformation in which several soft arc consistency operations can be applied simultaneously.

Definition 6.1 *Given a VCSP \mathcal{P} , a **Soft Arc Consistency (SAC) Transformation** is a set of $\mathbb{R} \cup \{\infty\}$ -closed **Project** and $\mathbb{R} \cup \{\infty\}$ -closed **UnaryProject** operations which, when applied simultaneously, transform \mathcal{P} into an equivalent valid VCSP. A SAC transformation is called **optimal** if it maximizes the lower bound ϕ_0 .*

If the operations which make up a SAC transformation are applied sequentially then the intermediate problems may have negative costs, since we only impose the condition that they lie in $\mathbb{R} \cup \{\infty\}$. The final result must nevertheless be a valid VCSP with nonnegative costs. In general, a SAC transformation may be applicable to a VCSP and increase ϕ_0 even when no individual SAC operation can be applied. (Recall that a SAC operation produces nonnegative costs). An example is given in [16]. However, this is not the case if the cost functions of \mathcal{P} are submodular, since Theorem 5.4 tells us that a path EPT can always be applied to \mathcal{P} , unless $\phi_0 = M = \min \text{Cost}_{\mathcal{P}}$, and a path EPT is simply a sequence of SAC operations.

Theorem 6.2 *Let \mathcal{P} be a submodular VCSP such that \mathcal{P} contains a nullary and all unary constraints (which may, in fact, be identically zero). Let M denote $\min \text{Cost}_{\mathcal{P}}$. Then there is an equivalent submodular VCSP \mathcal{P}' with*

1
2
3
4
5
6
7
8
9
10 *constraints on the same scopes as the constraints of \mathcal{P} and with an explicit*
11 *nullary constraint $\phi_0 = M$.*

12 **Proof:** It was shown in [16] that an optimal SAC transformation can be
13 found, for any VCSP, by first propagating infinite costs using a standard
14 generalized arc consistency algorithm [43, 40] and then by solving a linear
15 program. (A more detailed description of this algorithm is given below). If
16 the cost functions of \mathcal{P} are submodular, then an optimal SAC transformation
17 necessarily produces a VCSP \mathcal{P}' in which $\phi_0 = M$. To see this, suppose that
18 we had $\phi_0 < M$ in \mathcal{P}' , then, by Theorem 5.4, a path EPT (which is a form
19 of SAC transformation) would exist to increase ϕ_0 , but then this contradicts
20 the optimality of the SAC transformation. ■

21
22
23
24 We can therefore consider Optimal Soft Arc Consistency (OSAC), which
25 consists in finding and applying an optimal SAC transformation [16], as an
26 algorithm for minimizing locally-defined submodular functions. The linear
27 program has $O(ed+n)$ variables, where e is the number of valued constraints
28 in \mathcal{P} , d is the domain size and n is the number of variables in \mathcal{P} . In prac-
29 tice, for locally-defined SFM, this linear programming approach may very
30 well outperform combinatorial polynomial-time algorithms for submodular
31 function minimization [29, 31, 30, 50] despite the theoretical superiority of
32 these latter algorithms. Further research is required to compare, from a
33 theoretical and a practical point of view, our linear programming approach
34 with that of Cunningham [21] who uses Edmonds' greedy algorithm to solve
35 in pseudo-polynomial time a similar but much larger linear program.

36
37
38
39 The linear programming approach for finding an optimal SAC transfor-
40 mation was first published in Russian by Schlesinger in 1976 [49], but has
41 only recently appeared in English [16, 52]. The dual problem, which can
42 be considered as a linear relaxation of the original VCSP \mathcal{P} , has also been
43 studied independently, notably by Koster [37, 38]. We now give formally
44 the linear program and its dual.

45
46 We assume that all infinite costs have been propagated by SAC opera-
47 tions. This necessarily converges since each cost $\phi_\sigma(x)$ can be increased to
48 ∞ at most once. In fact, the propagation of infinite costs by SAC opera-
49 tions in the VCSP $\mathcal{P} = \langle \mathcal{V}, \mathcal{D}, \mathcal{C} \rangle$ can be achieved by establishing generalized
50 arc consistency [43] in the feasibility CSP $\text{Feas}(\mathcal{P}) = \langle V, D, \{ \langle \sigma, \text{Feas}(\phi) \rangle : \langle \sigma, \phi \rangle \in C \} \rangle$, a standard operation in Constraint Programming for which
51 efficient algorithms have been developed [40]. Alternatively, the constraints
52 of $\text{Feas}(\mathcal{P})$ can be decomposed into their binary projections and a standard
53 arc consistency algorithm applied [1].

$$\begin{array}{ll}
\text{(a)} & \text{Maximize}_{p,u} \sum_{i \in N} u_i \\
\text{(b)} & \text{such that } p_{i,a}^\sigma \in \mathbb{R} \quad (\sigma \in E, i \in N, a \in D) \\
\text{(c)} & u_i \in \mathbb{R} \quad (i \in N) \\
\text{(d)} & -\sum_{(\sigma \in E: i \in \sigma)} p_{i,a}^\sigma + u_i \leq \phi_i(a) \quad (i \in N, a \in D) \\
\text{(e)} & \sum_{i \in \sigma} p_{i,\bar{x}[i]}^\sigma \leq \phi_\sigma(\bar{x}) \quad (\sigma \in E, \bar{x} \in D^\sigma) \\
\text{(a')} & \text{Minimize}_\alpha \sum_{i \in N} \sum_{a \in D} \alpha_i(a) \phi_i(a) + \sum_{\sigma \in E} \sum_{\bar{x} \in D^\sigma} \alpha_\sigma(\bar{x}) \phi_\sigma(\bar{x}) \\
\text{(b')} & \text{such that } -\alpha_i(a) + \sum_{(\bar{x} \in D^\sigma, \bar{x}[i]=a)} \alpha_\sigma(\bar{x}) = 0 \quad (\sigma \in E, i \in N, a \in D) \\
\text{(c')} & \sum_{a \in D} \alpha_i(a) = 1 \quad (i \in N) \\
\text{(d')} & \alpha_i(a) \geq 0 \quad (i \in N, a \in D) \\
\text{(e')} & \alpha_\sigma(\bar{x}) \geq 0 \quad (\sigma \in E, \bar{x} \in D^\sigma)
\end{array}$$

Figure 9: The linear program to maximize the lower bound ϕ_0 of a VCSP (lines (a)-(e)) and its dual (lines (a')-(e'))

Then only finite costs can be propagated by **Project** and **UnaryProject**. We want to determine a set of such finite propagations which, when applied simultaneously, maximize the increase in ϕ_0 . For each valued constraint $\langle \sigma, \phi_\sigma \rangle$ of \mathcal{P} with $|\sigma| > 1$, and for each variable $i \in \sigma$, let $p_{i,a}^\sigma$ be the amount of cost projected by **Project** from ϕ_σ to $\phi_i(a)$. Costs moved from $\phi_i(a)$ to ϕ_σ are counted as negative in **Project**. Let u_i be the amount of cost projected by **UnaryProject** from ϕ_i to ϕ_0 . Then the problem is to maximize $\sum_i u_i$ while keeping the VCSP valid (i.e. no negative costs appear, thus ensuring that ϕ_0 is a lower bound). The resulting linear program is given in Figure 9 (lines (a)-(e)), together with its dual (lines (a')-(e')). The set of constraint scopes is denoted by E , the set of variables by $N = \{1, \dots, n\}$ and the domain of each variable by D .

The dual problem can be considered as a linear relaxation of the original VCSP \mathcal{P} [37, 38]. There is a one-to-one correspondence between $\{0, 1\}$ -valued solutions to the dual problem and solutions $\bar{x} \in D^n$ to the original VCSP \mathcal{P} , this correspondence being given by $\alpha_i(a) = 1$ iff $\bar{x}[i] = a$. Furthermore, the value of optimal solutions to the linear program (and its dual) is M , which is the value attained by optimal solutions \bar{x}^* to \mathcal{P} . Therefore the dual problem has $\{0, 1\}$ -valued optimal solutions, corresponding to the optimal solutions \bar{x}^* to the original VCSP \mathcal{P} . This result can be found in [52] for VCSPs with finite-valued binary constraints and a similar result is well known [21] for finite-valued VCSPs with a single global constraint.

7 Discussion

We have studied submodular function minimization (SFM) when the objective function is the sum of submodular cost functions whose arities are bounded by a constant. Cost functions may take on both positive real and infinite values.

We introduced the notion of a submodularity-preserving EPT and gave some examples. In fact, any transformation consisting of soft arc consistency operations [17, 16] is a submodularity-preserving EPT. An intriguing open question is whether any other equivalence-preserving transformations exist which preserve submodularity (or other multimorphisms [11]).

The main result of this paper is that for any problem consisting of the minimization of the sum of submodular functions, there always exists an equivalent problem with cost functions on the same set of scopes and with the actual minimum value of the objective function rendered explicit as a nullary constraint. This equivalent problem can be found by establishing optimal soft arc consistency which can be achieved by solving a linear program. Further research is required to determine whether this provides us with an algorithm which is theoretically and/or practically efficient.

Since this linear program makes no explicit use of the domain orders, it also finds the minimum value of a locally-defined *permuted* submodular function, that is an objective function which can be made submodular by applying possibly different permutations to each variable domain. Schlesinger [48] showed that given a locally-defined finite-valued permuted submodular function, domain permutations which render it submodular can be found in polynomial time by reduction to 2SAT.

Fujishige et al. [25] have recently devised and tested a minimum-norm-point SFM algorithm based on Wolfe's algorithm [53]. Although it is an open problem to determine whether this algorithm is polynomial or not, it proved dramatically faster than polynomial-time SFM algorithms [50, 31, 30] on two classes of binary submodular VCSPs over boolean domains. Further trials are clearly necessary to compare experimentally the various algorithms now at our disposal for solving locally-defined SFM with non-binary valued constraints over non-boolean domains.

Submodularity has recently been generalized to a larger tractable class in which the functions min,max are replaced by arbitrary conservative commutative functions [13]. (A function $f : D^2 \rightarrow D$ is conservative if $\forall a, b \in D, f(a, b) \in \{a, b\}$). Any algorithm for SFM could directly be applied here, since the solution technique for this new tractable class is to eliminate unnecessary values from domains, then apply an SFM algorithm after domain

1
2
3
4
5
6
7
8
9
10 re-ordering.

11 In the theory of tractable constraint classes, some form of consistency is
12 a solution procedure for the majority of the known tractable classes. This
13 paper shows that this is again the case for the class of submodular valued
14 constraints, since optimal soft arc consistency is a solution procedure.
15

16 8 Acknowledgments

17
18 The research reported in this paper would not have been possible without
19 valuable discussions with David Cohen, Simon de Givry, Peter Jeavons and
20 Andrei Krokhin. I would also like to thank the anonymous referees whose
21 comments greatly improved the presentation of this paper.
22
23
24

25 References

- 26
27
28 [1] Bessière, C. & Régin, J.-C. “Refining the basic constraint propagation
29 algorithm”, *Proc IJCAI’01*, Seattle WA (2001) pp. 309–315.
30
31 [2] Billionet, A. & Minoux, M. “Maximizing a supermodular pseudo-
32 boolean function: a polynomial algorithm for cubic functions”, *Dis-
33 crete Applied Mathematics* 12 (1985) pp. 1–11.
34
35 [3] Boros, E. & Hammer, P.L. “Pseudo-boolean optimization”, *Discrete
36 Applied Mathematics* 123 (2002) pp. 155–225.
37
38 [4] Bulatov, A.A. “A dichotomy theorem for constraints on a three-
39 element set”, *Proc. 43rd IEEE Symposium on Foundations of Com-
40 puter Science (FOCS’02)*, (2002) pp. 649–658.
41
42 [5] Bulatov, A.A. “Tractable conservative constraint satisfaction prob-
43 lems”, *Proc. 18th IEEE Symposium on Logic in Computer Science
44 (LICS’03)*, (2003) pp. 321–330.
45
46 [6] Bulatov, A.A. “Combinatorial problems raised from 2-semilattices”,
47 *Journal of Algebra*, (2006) pp. 321–339.
48
49 [7] Bulatov, A.A., Krokhin, A.A. & Jeavons, P.G. “Classifying the com-
50 plexity of constraints using finite algebras” *SIAM Journal on Com-
51 puting*, 34 (2005) pp. 720–742.
52
53
54
55
56
57
58
59
60
61
62
63
64
65

- 1
2
3
4
5
6
7
8
9
10 [8] Cohen, D., Cooper, M.C. & Jeavons, P. “A complete characterisation
11 of complexity for Boolean constraint optimization problems”, *Proc.*
12 *10th Int. Conf. on Principles and Practice of Constraint Programming*
13 *(CP’04)*, LNCS 3258 (2004) pp. 212–226.
- 14
15 [9] Cohen, D., Cooper, M.C., Jeavons, P. & Krokhin, A. “A maximal
16 tractable class of soft constraints”, *Journal of Artificial Intelligence*
17 *Research* 22 (2004) pp. 1–22.
- 18
19 [10] Cohen, D., Cooper, M.C., Jeavons, P. & Krokhin, A. “Supermodular
20 functions and the complexity of Max-CSP” *Discrete Applied Mathe-*
21 *matics* 149 (2005) pp. 53–72.
- 22
23 [11] Cohen, D., Cooper, M.C., Jeavons, P. & Krokhin, A. “The complexity
24 of soft constraint satisfaction”, *Artificial Intelligence* 170(11) (2006)
25 pp. 983–1016.
- 26
27 [12] Cohen, D., Cooper, M.C. & Jeavons, P. “An algebraic characterisation
28 of complexity for valued constraints”, *Proc. CP’06*, LNCS 4204 (2006)
29 pp. 107–121.
- 30
31 [13] Cohen, D., Cooper, M.C. & Jeavons, P. “Generalising Submodular-
32 ity and Horn Clauses: Tractable Optimization Problems Defined by
33 Tournament Pair Multimorphisms”, to appear in *Theoretical Com-*
34 *puter Science* (2008).
- 35
36 [14] Cooper, M.C. “Reduction operations in fuzzy and valued constraint
37 satisfaction”, *Fuzzy Sets and Systems* 134 (2003) pp. 311–342.
- 38
39 [15] Cooper, M.C. “High-order consistency in valued constraint satisfac-
40 tion”, *Constraints* 10 (2005) pp. 283–305.
- 41
42 [16] Cooper, M.C., de Givry, S. & Schiex, T. “Optimal soft arc consis-
43 tency”, *Proc. IJCAI’07*, Hyderabad (2007) pp. 68–73.
- 44
45 [17] Cooper, M.C. & Schiex, T. “Arc consistency for soft constraints”,
46 *Artificial Intelligence* 154(1-2) (2004) pp. 199–227.
- 47
48 [18] Creignou, N. “A dichotomy theorem for maximum generalised satisfi-
49 ability problems”, *Journal of Computer and Systems Sciences* 51(3),
50 (1995) pp. 511–522.
- 51
52 [19] Creignou, N., Khanna, S. & Sudan, M. *Complexity classification of*
53 *Boolean constraint satisfaction problems*, volume 7 of *SIAM Mono-*
54 *graphs on Discrete Mathematics and Applications* (2001).
- 55
56
57
58
59
60
61
62
63
64
65

- 1
2
3
4
5
6
7
8
9
10 [20] Cunningham, W.H. “Minimum cuts, modular functions, and matroid
11 polyhedra”, *Networks* 15(2) (1985) pp. 205–215.
12
13 [21] Cunningham, W.H. “On submodular function minimization” *Combi-*
14 *natorica* 5 (1985) pp. 185–192.
15
16 [22] Dechter, R. *Constraint Processing*, Morgan Kaufmann, 2003.
17
18 [23] Feder, T. & Vardi, M.Y. “The computational structure of monotone
19 monadic SNP and constraint satisfaction: a study through datalog and
20 group theory”, *SIAM Journal on Computing* 28(1), (1998) pp. 57–104.
21
22 [24] Fujishige, S., *Submodular Functions and Optimisation*, 2nd edn., An-
23 nals of Discrete Mathematics 58, Elsevier, 2005.
24
25 [25] Fujishige, S., Hayashi, T. & Isotani, S. “The minimum-norm-point al-
26 gorithm applied to submodular function minimization and linear pro-
27 gramming”, Report RIMS1571, Research Institute for Mathematical
28 Sciences, Kyoto University (2006).
29
30 [26] Fujishige, S. & Patkar, S.B. “Realization of set functions as cut func-
31 tions of graphs and hypergraphs”, *Discrete Mathematics* 226 (2001)
32 pp. 199–210.
33
34 [27] Goldberg, A. & Tarjan, R.E. “A new approach to the maximum flow
35 problem”, *Journal of the ACM* 35 (1988) pp. 921–940.
36
37 [28] Grötschel, M., Lovász, L. & Schrijver, A. “The ellipsoid method and its
38 consequences in combinatorial optimization”, *Combinatorica* 1 (1981)
39 pp. 169–198; corrigendum: *Combinatorica* 4 (1984) pp. 291–295.
40
41 [29] Iwata, S. “A fully combinatorial algorithm for submodular func-
42 tion minimization”, *Journal of Combinatorial Theory, Series B* 84(2)
43 (2002) pp. 203–212.
44
45 [30] Iwata, S. “A faster scaling algorithm for minimizing submodular func-
46 tions”, *SIAM J. Comput.* 32(4) (2003) pp. 833–840.
47
48 [31] Iwata, S., Fleischer, L. & Fujishige, S. “A combinatorial, strongly
49 polynomial-time algorithm for minimizing submodular functions”,
50 *Journal of the ACM* 48(4), (2001), pp. 761–777.
51
52 [32] Jeavons, P.G. “On the algebraic structure of combinatorial problems”,
53 *Theoretical Computer Science* 200 (1998) pp. 185–204.
54
55
56
57
58
59
60
61
62
63
64
65

- 1
2
3
4
5
6
7
8
9
10 [33] Jeavons, P., Cohen, D. & Cooper M.C. “Constraints, consistency and
11 closure”, *Artificial Intelligence* 101 (1998) pp. 251–265.
12
13 [34] Jeavons, P.G., Cohen D.A. & Gyssens, M. “Closure properties of con-
14 straints”, *Journal of the ACM* 44 (1997) pp. 527–548.
15
16 [35] Jeavons, P.G. & Cooper, M.C. “Tractable constraints on ordered do-
17 mains”, *Artificial Intelligence* 79 (2), (1995) pp. 327–339.
18
19 [36] Jonsson, P., Klasson, M. & Krokhin, A. “The approximability of
20 three-valued MAX CSP”, *SIAM Journal on Computing* 35 (6), (2006)
21 pp. 1329–1349.
22
23 [37] Koster, A. *Frequency Assignment - Models and Algorithms*, Ph.D.
24 thesis, Universiteit Maastricht, The Netherlands (1999), ISBN 90-
25 9013119-1.
26
27 [38] Koster, A.M.C.A., van Hoesel, S.P.M. & Kolen, A.W.J. “The partial
28 constraint satisfaction problem: facets and lifting theorems”, *Oper.*
29 *Res. Lett.* 23 (3-5) (1998) pp. 89–97.
30
31 [39] Larrosa, J. & Schiex, T. “Solving weighted CSP by maintaining arc
32 consistency”, *Artificial Intelligence* 159 (2004) pp. 1–26.
33
34 [40] Lecoutre, C. & Szymanek, R. “Generalized arc consistency for posi-
35 tive table constraints” in *Proc. Principles and Practice of Constraint*
36 *Programming - CP 2006*, Benhamou, F. (ed.), LNCS 4204, Springer-
37 Verlag, Berlin (2006) pp. 284–298.
38
39 [41] McCormick, S.T. “Submodular function minimization in discrete op-
40 timization” in *Handbooks in Operations Research and Management*
41 *Science* 12, Aardal, K., Nemhauser, G.L. & Weismantel, R. (eds.),
42 Elsevier, Amsterdam (2005) pp. 321–391.
43
44 [42] Meseguer, P., Rossi, F. & Schiex, T. “Soft Constraints” in *Handbook*
45 *of Constraint Programming*, eds. Rossi, F., van Beek, P. & Walsh, T.,
46 Elsevier (2006) pp. 281–328.
47
48 [43] Mohr, R. & Masini, G. “Good old discrete relaxation”, *Proc. European*
49 *Conf. Artificial Intelligence*, Munich (1988) pp. 651–656.
50
51 [44] Narayanan, H, *Submodular Functions and Electrical Networks*, North-
52 Holland, Amsterdam (1997).
53
54
55
56
57
58
59
60
61
62
63
64
65

- 1
2
3
4
5
6
7
8
9
10 [45] Orlin, J.B., “A faster strongly polynomial time algorithm for submodular minimization”, *IPCO 2007*, Fischetti, M. & Williamson, D.P. (Eds), *LNCS* 4513 (2007) pp. 240–251.
11
12
13
14 [46] Queyranne, M. “Minimising symmetric submodular functions”, *Mathematical Programming* 82 (1-2) (1998) pp. 3-12.
15
16
17 [47] Schiex, T., Fargier, H. & Verfaillie, G. “Valued constraint satisfaction problems: hard and easy problems”, *Proc. of the 14th IJCAI*, Montreal, Canada (1995) pp. 631–637.
18
19
20
21 [48] Schlesinger, D. “Exact solution of permuted submodular MinSum problems”, to appear in *Proc. of the 6th International Conference on Energy Minimization Methods in Computer Vision and Pattern Recognition*, Ezhou, Hubei, China (2007).
22
23
24
25
26
27 [49] Schlesinger, M. “Syntactic analysis of two-dimensional visual signals in noisy conditions” (In Russian), *Kibernetika* 4 (1976) pp. 113–130.
28
29
30 [50] Schrijver, A. “A combinatorial algorithm minimizing submodular functions in strongly polynomial time”, *Journal of Combinatorial Theory*, Ser. B, 80 (2000) pp. 346–355.
31
32
33
34 [51] Topkis, D. *Supermodularity and Complementarity*, Princeton University Press (1998).
35
36
37 [52] Werner, T. “A Linear Programming Approach to Max-sum Problem: A Review”, *IEEE Trans. on Pattern Analysis and Machine Intelligence* 29(7) (2007) pp. 1165–1179.
38
39
40
41
42 [53] Wolfe, P. “Finding the nearest point in a polytope” *Mathematical Programming* 11 (1976) pp. 128–149.
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65