Efficient Algorithms For Constraint Description Problems Over Finite Totally Ordered Domains* (Extended Abstract)

Ángel J. Gil[1], Miki Hermann[2], Gernot Salzer[3], and Bruno Zanuttini[4]

[1] Universitat Pompeu Fabra, Barcelona, Spain. angel.gil@upf.edu
[2] LIX (FRE 2653), École Polytechnique, France. hermann@lix.polytechnique.fr
[3] Technische Universität Wien, Austria. salzer@logic.at
[4] GREYC (UMR 6072), Université de Caen, France. zanutti@info.unicaen.fr

Abstract. Given a finite set of vectors over a finite totally ordered domain, we study the problem of computing a constraint in conjunctive normal form such that the set of solutions for the produced constraint is identical to the original set. We develop an efficient polynomial-time algorithm for the general case, followed by specific polynomial-time algorithms producing Horn, dual Horn, and bijunctive constraints for sets of vectors closed under the operations of conjunction, disjunction, and median, respectively. We also consider the affine constraints, analyzing them by means of computer algebra. Our results generalize the work of Dechter and Pearl on relational data, as well as the papers by Hébrard and Zanuttini. They also complete the results of Hähnle *et al.* on multi-valued logics and Jeavons *et al.* on the algebraic approach to constraints. We view our work as a step toward a complete complexity classification of constraint satisfaction problems over finite domains.

1 Introduction and Summary of Results

Constraint satisfaction problems constitute nowadays a well-studied topic on the frontier of complexity, logic, combinatoric, and artificial intelligence. It is indeed well-known that this framework allows us to encode many natural problems or knowledge bases. In principle, an instance of a constraint satisfaction problem is a finite set of variable vectors associated with an allowed set of values. A *model* is an assignment of values to all variables that satisfy every constraint. When a constraint satisfaction problem encodes a decision problem, the models represent its *solutions*. When it encodes some knowledge, the models represent possible combinations that the variables can assume in the described universe.

The constraints are usually represented by means of a set of variable vectors associated with an allowed set of values. This representation is not always well-suited for our purposes, therefore other representations have been introduced. The essence of the most studied alternative is the notion of of a *relation*, making it easy to apply it within the database or knowledge base framework. An

* Dedicated to the memory of Peter Ružička (1947 – 2003).

instance of a constraint satisfaction problem is then represented as a conjunction of relation applications. We study in this paper the constraint *description* problem, i.e., that of converting a constraint from a former set representation to the latter one by means of conjunction over relations. We consider this problem first in its general setting without any restrictions imposed on the initial set of vectors. We continue by imposing several closure properties on the initial set, like the closure under the minimum, maximum, median, and affine operations. We subsequently discover that these closure properties induce the description by Horn, dual Horn, bijunctive, and affine constraints, respectively.

The motivation to study constraint description problems is numerous. From the artificial intelligence point of view description problems formalize the notion of exact acquisition of knowledge from examples. This means that they formalize situations where a system is given access to a set of examples and it is asked to compute a constraint describing it exactly. Satisfiability poses a keystone problem in artificial intelligence, automated deduction, databases, and verification. It is well-known that the satisfiability problem for the general constraints is an NP-complete problem. Therefore it is important to look for restricted classes of constraints that admit polynomial algorithms deciding satisfiability. Horn, dual Horn, bijunctive, and affine constraints constitute exactly these tractable classes, as it was mentioned by Schaefer [17] for the case of Boolean constraints. Thus the description problem for these four classes can be seen as storing a specific knowledge into a knowledge base while we are required to respect its format. This problem is also known as *structure identification*, studied by Dechter with Pearl [7] and by Hébrard with Zanuttini [11, 19], both for the Boolean case. Another motivation for studying description problems comes from combinatorics. Indeed, since finding a solution for an instance of a constraint satisfaction problem is difficult in general but tractable in the four aforementioned cases, it is important to be able to recognize constraints belonging to these tractable cases.

The study of Boolean constraint satisfaction problems, especially their complexity questions, was started by Schaefer in [17], although he did not yet consider constraints explicitly. During the last ten years, constraints gained considerable interest in theoretical computer science. An excellent complexity classification of existing Boolean constraint satisfaction problems can be found in the monograph [6]. Jeavons *et al.* [5, 13, 14] started to study constraint satisfaction problems from an algebraic viewpoint. Feder, Kolaitis, and Vardi [8, 16] posed a general framework for the study of constraint satisfaction problems.

There has not been much progress done on constraint satisfaction problems on domains with larger cardinality. Hell and Nešetřil [12] studied constraint satisfaction problems by means of graph homomorphisms. Bulatov [4] made a significant breakthrough with a generalization of Schaefer's result to a three-element domain. On the other hand, Hähnle *et al.* [2, 3, 10] studied the complexity of satisfiability problems for many-valued logics that present yet another viewpoint of constraint satisfaction problems. We realized reading the previous articles on many-valued logics that in the presence of a total order the satisfiability problem for the Horn, dual Horn, bijunctive, and affine many-valued formulas of signed

logic are decidable in polynomial time. We saw also that Jeavons and Cooper [15] studied some aspects of tractable constraints on finite ordered domains from an algebraic standpoint. This lead us to the idea to look more carefully on constraint description problems over finite totally ordered domains, developing a new formalism for constraints based on an already known concept of inequalities. The purpose of our paper is manifold. We want to generalize the work of Dechter and Pearl [7], based on the more efficient algorithms for Boolean description problems by Hébrard and Zanuttini [11, 19]. We also want to complement the work of Hähnle *et al.* on many-valued logics. We also want to pave the way for a complete complexity classification of constraint satisfaction problems over finite totally ordered domains[5].

2 Preliminaries

Let D be a finite, totally ordered domain, say $D = \{0, \ldots, n-1\}$, and let V be a set of variables. For $x \in V$ and $d \in D$, the inequalities $x \geq d$ and $x \leq d$ are called positive and negative *literal*, respectively. The set of *constraints* over D and V is defined as follows: the logical constants *false* and *true* are constraints; literals are constraints; if φ and ψ are constraints, then the expressions $(\varphi \wedge \psi)$ and $(\varphi \vee \psi)$ are constraints. We write $\varphi(x_1, \ldots, x_\ell)$ to indicate that constraint φ contains exactly the variables $x_1, \ldots, x_\ell$. For convenience, we use the following shorthand notation, as usual: $x > d$ means $x \geq d+1$ for $d \in \{0, \ldots, n-2\}$, and *false* otherwise; $x < d$ means $x \leq d-1$ for $d \in \{1, \ldots, n-1\}$, and *false* otherwise; $x = d$ means $x \geq d \wedge x \leq d$; $\neg$*false* and $\neg$*true* mean *true* and *false*, respectively; $\neg(x \geq d)$, $\neg(x \leq d)$, $\neg(x > d)$, and $\neg(x < d)$ mean $x < d$, $x > d$, $x \leq d$, and $x \geq d$, respectively; $\neg(x = d)$ and $x \neq d$ both mean $x < d \vee x > d$; $\neg(\varphi \wedge \psi)$ and $\neg(\varphi \vee \psi)$ mean $\neg\varphi \vee \neg\psi$ and $\neg\varphi \wedge \neg\psi$, respectively. Note that $x = d$ and $x \neq d$ asymptotically require the same space as their alternative notations, i.e., $O(\log n)$. Since d is bounded by n, its binary coding has length $O(\log n)$.

A *clause* is a disjunction of literals. It is a *Horn* clause if it contains at most one positive literal, *dual Horn* if it contains at most one negative literal, and *bijunctive* if it contains at most two literals. Following Schaefer [17], we extend the notion of constraints by introducing affine clauses. An *affine* clause is an equation $a_1x_1 + \cdots + a_\ell x_\ell = b \pmod{n}$ where $x_1, \ldots, x_\ell \in V$ and $a_1, \ldots, a_\ell, b \in D$. A constraint is in *conjunctive normal form* (CNF) if it is a conjunction of clauses. It is a Horn, a dual Horn, a bijunctive, or an affine constraint if it is a conjunction of Horn, dual Horn, bijunctive, or affine clauses, respectively.

A *model* for a constraint $\varphi(x_1, \ldots, x_\ell)$ is a mapping $m\colon \{x_1, \ldots, x_\ell\} \to D$ assigning a domain element $m(x)$ to each variable x. The *satisfaction relation* $m \models \varphi$ is inductively defined as follows: $m \models$ *true* and $m \not\models$ *false*; $m \models x \leq d$ if $m(x) \leq d$, and $m \models x \geq d$ if $m(x) \geq d$; $m \models \varphi \wedge \psi$ if $m \models \varphi$ and $m \models \psi$; $m \models \varphi \vee \psi$ if $m \models \varphi$ or $m \models \psi$. An affine clause is satisfied by a model m if $a_1m(x_1) + \cdots + a_\ell m(x_\ell) = b \pmod{n}$. The set of all models satisfying φ is denoted

[5] See `http://www.lix.polytechnique.fr/~hermann/publications/cdesc04.ps.gz` for the proofs, since many of them are omitted here due to lack of space.

by $\mathrm{Sol}(\varphi)$. If we arrange the variables in some arbitrary but fixed order, say as a vector $x = (x_1, \ldots, x_\ell)$, then the models can be identified with the vectors in D^ℓ. The j-th component of a vector m, denoted by $m[j]$, gives the value of the j-th variable, i.e., $m(x_j) = m[j]$. The operations of conjunction, disjunction, addition, and median on vectors $m, m', m'' \in D^\ell$ are defined as follows:

$$\begin{aligned} m \wedge m' &= (\min(m[1], m'[1]), \ldots, \min(m[\ell], m'[\ell])) \\ m \vee m' &= (\max(m[1], m'[1]), \ldots, \max(m[\ell], m'[\ell])) \\ m + m' &= (m[1] + m'[1] \pmod{|D|}, \ldots, m[\ell] + m'[\ell] \pmod{|D|}) \\ \mathrm{med}(m, m', m'') &= (\mathrm{med}(m[1], m'[1], m''[1]), \ldots, \mathrm{med}(m[\ell], m'[\ell], m''[\ell]) \end{aligned}$$

The ternary *median* operator is defined as follows: for each choice of three values $a, b, c \in D$ such that $a \leq b \leq c$, we have $\mathrm{med}(a, b, c) = b$. Note that the median can also be defined by $\mathrm{med}(a, b, c) = \min(\max(a, b), \max(b, c), \max(c, a))$.

We say that a set of vectors is *Horn* if it is closed under conjunction, *dual Horn* if it is closed under disjunction, *bijunctive* if it is closed under median, and *affine* if it is a Cartesian product of affine spaces, i.e., of vector spaces translated by some vector.

3 Constraints in Conjunctive Normal Form

We investigate first the description problem for arbitrary sets of vectors.

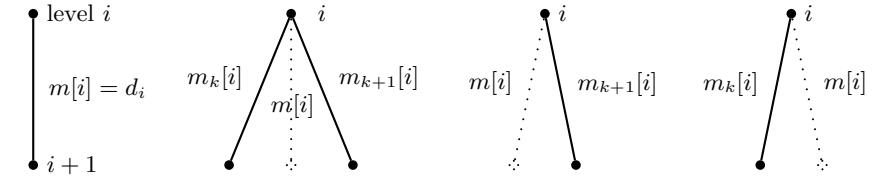
Problem: DESCRIPTION
Input: A finite set of vectors $M \subseteq D^\ell$ over a finite totally ordered domain D.
Output: A constraint $\varphi(x_1, \ldots, x_\ell)$ over D in CNF such that $\mathrm{Sol}(\varphi) = M$.

The usual approach to this problem in the literature is to compute first the complement set $\bar{M} = D^\ell \setminus M$, followed by a construction of a clause $c(\bar{m})$ for each vector $\bar{m} \in \bar{M}$ missing from M such that $\bar{m}$ is the unique vector falsifying $c(\bar{m})$. The constraint φ is then the conjunction of the clauses $c(\bar{m})$ for all missing vectors $\bar{m} \in \bar{M}$. However, this algorithm is essentially exponential, since the complement set $\bar{M}$ can be exponentially bigger than the original set of vectors M. We present a new algorithm running in polynomial time and producing a CNF constraint of polynomial length with respect to $|M|$, ℓ, and $\log |D|$.

To construct the constraint φ we proceed in the following way. We arrange the set M as an ordered tree T_M, with branches corresponding to the vectors in M. In case M contains all possible vectors, i.e. $M = D^\ell$, T_M is a complete tree of branching factor $|D|$ and depth ℓ. Otherwise, some branches are missing, leading to gaps in the tree. We characterize these gaps by conjunctions of literals. Their disjunction yields a complete description of all vectors that are *missing* from M. Finally, by negation and de Morgan's law we obtain φ.

Let T_M be an ordered tree with edges labeled by domain elements such that each path from the root to a leaf corresponds to a vector in M. The tree T_M contains a path labeled $d_1. \cdots .d_i$ from the root to some node if there is a vector

$m \in M$ such that $m[j] = d_j$ holds for every $j = 1, \ldots, i$. The level of a node is its distance to the root plus 1, i.e., the root is at level 1 and a node reachable via $d_1.\cdots.d_i$ is at level $i+1$ (Fig. 1(a)). Note that all leaves are at level $\ell+1$. If the edges between a node and its children are sorted in ascending order according to their labels, then traversing the leaves from left to right enumerates the vectors of M in lexicographic order, say $m_1, \ldots, m_{|M|}$. A vector m is lexicographically smaller than a vector m', if there is a level i such that $m[i] < m'[i]$ holds, and for all $j < i$ we have $m[j] = m'[j]$.



(a) Level i (b) Missing between (c) ... left (d) ... right

Fig. 1. Tree representation of vectors

Suppose that m_k and m_{k+1} are immediate neighbors in the lexicographic enumeration of M and let m be a vector lexicographically in between, thus missing from M. There are three possibilities for the path corresponding to m. It may either leave the tree at the fork *between* m_k and m_{k+1} (Fig. 1(b)), or at the fork to the *left* of m_{k+1} (Fig. 1(c)), or at the fork to the *right* of m_k (Fig. 1(d)). Therefore the missing vector m can be characterized by the constraints middle(k, i), left$(k+1, i)$, and right(k, i) defined as follows.

$$\begin{aligned}
\text{middle}(k,i) &= \bigwedge_{j<i}(x_j = m_k[j]) \quad \wedge \quad (x_i > m_k[i]) \ \wedge \ (x_i < m_{k+1}[i]) \\
\text{left}(k+1,i) &= \bigwedge_{j<i}(x_j = m_{k+1}[j]) \quad \wedge \quad (x_i < m_{k+1}[i]) \\
\text{right}(k,i) &= \bigwedge_{j<i}(x_j = m_k[j]) \quad \wedge \quad (x_i > m_k[i])
\end{aligned}$$

The situation depicted in Fig. 1 is a snapshot at level i of the tree T_M. Of course, if the situations in Fig. 1(c) or Fig. 1(d) occur at a level i, then subsequent forks of a missing vector to the left or to the right, respectively, occur also at each level $i' > i$.

To describe *all* vectors missing from M we form the disjunction of the above constraints for appropriate values of k and i. We need to determine the levels at which neighboring models fork by means of the following function.

$$\text{fork}(k) = \begin{cases} 0 & \text{for } k = 0 \\ \min\{i \mid m_k[i] \neq m_{k+1}[i]\} & \text{for } k = 1, \ldots, |M|-1 \\ 0 & \text{for } k = |M| \end{cases}$$

The values fork(0) and fork($|M|$) correspond to imaginary models m_0 and $m_{|M|+1}$ forking at a level above the root. They allow to write the conditions below in a concise way at the left and right border of the tree. The three situations in Fig. 1 can now be specified by the following conditions.

$$\begin{array}{lcll} i = \mathrm{fork}(k) & \wedge & m_k[i] + 1 < m_{k+1}[i] & \text{(edges missing in between)} \\ \mathrm{fork}(k) < i & \wedge & m_{k+1}[i] > 0 & (\ldots \text{ to the left}) \\ \mathrm{fork}(k) < i & \wedge & m_k[i] < |D| - 1 & (\ldots \text{ to the right}) \end{array}$$

The second condition in each line ensures that there is at least one missing edge. It avoids that the constraints middle(k, i), left$(k + 1, i)$, and right(k, i) evaluate to false. The disjunction of clauses middle(k, i), left$(k+1, i)$, and right(k, i) that satisfy the first, second, and third condition, respectively, for all models and all levels represent a disjunctive constraint satisfied by the models *missing* from M. After applying negation and de Morgan's law, we arrive at the constraint

$$\begin{aligned} &\varphi(M) \\ &\quad = \quad \bigwedge \{\, \neg\,\mathrm{middle}(k,i) \mid 0 < k < |M|\,,\ i = \mathrm{fork}(k),\ m_k[i] + 1 < m_{k+1}[i] \,\} \\ &\quad \wedge \quad \bigwedge \{\, \neg\,\mathrm{left}(k+1,i) \mid 0 \le k < |M|\,,\ \mathrm{fork}(k) < i \le \ell,\ m_{k+1}[i] > 0 \,\} \\ &\quad \wedge \quad \bigwedge \{\, \neg\,\mathrm{right}(k,i) \mid 0 < k \le |M|\,,\ \mathrm{fork}(k) < i \le \ell,\ m_k[i] < |D| - 1 \,\}. \end{aligned}$$

The constraint φ is in conjunctive normal form and the condition $\mathrm{Sol}(\varphi) = M$ holds. Note that we use negation not as an operator on the syntax level but as a meta-notation expressing that the constraint following the negation sign has to be replaced by its dual. Note also that the conjunct left$(k + 1, i)$ is defined and used with the shifted parameter $k + 1$. This is necessary because the conjunct left characterizes a gap lexicographically before the vector m_{k+1}.

It follows directly from the construction that the constraint $\varphi(M)$ contains at most $3\,|M|\,\ell$ clauses. Each clause contains at most 2ℓ literals, namely at most one positive and one negative for each variable. Each literal has the length $O(\log |D|)$, since the domain elements are written in binary notation. Hence, the overall length of the constraint $\varphi(M)$ is $O(|M|\,\ell^2 \log |D|)$.

The vectors in M can be lexicographically sorted in time $O(|M|\,\ell \log |D|)$ using radix sort. The factor $\log |D|$ stems from the comparison of domain elements. The fork levels can also be computed in time $O(|M|\,\ell \log |D|)$, in parallel with sorting the set M. The constraint $\varphi(M)$ is produced by two loops, where the outer loop is going through each vector in M and the inner loop through the variables. The clauses $\neg\,\mathrm{middle}(k, i)$, $\neg\,\mathrm{left}(k + 1, i)$, and $\neg\,\mathrm{right}(k, i)$ are potentially written in each step inside the combined loops. This makes an algorithm with time complexity $O(|M|\,\ell^2 \log |D|)$.

Theorem 1. *For each set of vectors $M \subseteq D^\ell$ over a finite ordered domain D there exists a constraint φ in conjunctive normal form such that $M = \mathrm{Sol}(\varphi)$. It contains at most $3\,|M|\,\ell$ clauses and its length is $O(|M|\,\ell^2 \log |D|)$. The algorithm constructing φ runs in time $O(|M|\,\ell^2 \log |D|)$ and space $O(|M|\,\ell \log |D|)$.*

4 Horn Constraints

Horn clauses and formulas constitute a frequently studied subclass. We investigate in this section the description problem for a generalization of Horn formulas to ordered finite domains, namely for sets of vectors closed under conjunction.

Problem: DESCRIPTION[HORN]
Input: A finite set of vectors $M \subseteq D^\ell$, closed under conjunction, over a finite totally ordered domain D.
Output: A Horn constraint φ over D such that $\mathrm{Sol}(\varphi) = M$.

The general construction in Section 3 does not guarantee that the final constraint is Horn whenever the set M is closed under conjunction. Therefore we must shorten the clauses of the constraint φ, produced in Section 3, to obtain only Horn clauses. For this, we will modify a construction proposed by Jeavons and Cooper in [15]. Their method is exponential, since it proposes to construct a Horn clause for each vector in the complement set $D^\ell \setminus M$. Contrary to the method of Jeavons and Cooper, our new proposed method is polynomial with respect to $|M|$, ℓ and $\log |D|$.

Let $\varphi(M)$ be a constraint produced by the method of Section 3 and let c be a clause from $\varphi(M)$. We denote by c^- the disjunction of the negative literals in c. The vectors in M satisfying a negative literal in c satisfy also the restricted clause c^-. Hence we have only to care about the vectors that satisfy a positive literal but no negative literals in c, described by the set

$$M_c \quad = \quad \{m \in M \mid m \not\models c^-\} \ .$$

If M_c is empty, we can replace the clause c by $h(c) = c^-$ in the constraint $\varphi(M)$ without changing the set of models $\mathrm{Sol}(\varphi)$. Otherwise, note that M_c is closed under conjunction, since M is already closed under this operation. Indeed, if the vectors m and m' falsify every negative literal $x \leq d$ of c^- then the conjunction $m \wedge m'$ falsifies the same negative literals. Hence M_c contains a unique minimal model $m_* = \bigwedge M_c$. Every positive literal in c satisfied by m_* is also satisfied by all the vectors in M_c. Let l be a positive literal from c and satisfied by m_*. There exists at least one such literal since otherwise m_* would satisfy neither c^- nor any positive literal in c, hence it would not be in M_c. Then c can be replaced with the Horn clause $h(c) = l \vee c^-$, without changing the set of models $\mathrm{Sol}(\varphi)$. We obtain a Horn constraint $h(M)$ for a Horn set M by replacing every non-Horn clause c in $\varphi(M)$ by its Horn restriction $h(c)$.

The length of $h(M)$ is basically the same as that of $\varphi(M)$. The number of clauses is the same and the length of clauses is $O(\ell \log |D|)$ in both cases. There are at most 2ℓ literals in each clause of $\varphi(M)$ (one positive and one negative literal per variable) versus $\ell + 1$ literals in each clause of $h(M)$ (one negative literal per variable plus a single positive literal).

The construction of each Horn clause $h(c)$ requires time $O(|M|\, \ell \log |D|)$. For every vector $m \in M$ we have to evaluate at most ℓ negative literals in c to find out whether m belongs to M_c. The evaluation of a literal takes time

$O(\log|D|)$. Hence the computation of the set M_c takes time $O(|M|\,\ell\log|D|)$. To obtain $m_* = \bigwedge M_c$, we have to compute $|M_c|-1$ conjunctions between vectors of length ℓ, each of the ℓ conjunctions taking time $O(\log|D|)$. Therefore m_* can also be computed in time $O(|M|\,\ell\log|D|)$. Since there are at most $3\,|M|\,\ell$ clauses in $\varphi(M)$, the transformation of $\varphi(M)$ into $h(M)$ can be done in time $O(|M|^2\,\ell^2\log|D|)$. Hence, the whole algorithm producing the Horn constraint $h(M)$ from the set of vectors M runs in time $O(|M|^2\,\ell^2\log|D|)$.

Theorem 2. *For each set of vectors $M \subseteq D^\ell$ over a finite totally ordered domain D that is closed under conjunction, there exists a Horn constraint φ such that $M = \mathrm{Sol}(\varphi)$. The constraint φ contains at most $3\,|M|\,\ell$ clauses and its length is $O(|M|\,\ell^2\log|D|)$. The algorithm constructing the constraint φ runs in time $O(|M|^2\,\ell^2\log|D|)$ and space $O(|M|\,\ell\log|D|)$.*

Note that the result of Theorem 2 is not optimal. We can indeed derive a better algorithm with time complexity $O(|M|\,\ell(|M|+\ell)\log|D|)$ from the one for the Boolean case presented in [11]. For that purpose, notice that the conjuncts $\mathrm{middle}(k,i)$ and $\mathrm{left}(k+1,i)$ behave like the first case studied in the aforementioned paper, whereas the conjunct $\mathrm{right}(k,i)$ can be treated by means of the other cases, taking advantage of the following sets defined as generalizations of the corresponding ones from [11]:

$$\begin{aligned} I(k,i) &= \{m \in M \mid m[i] > m_k[i] \text{ and } \forall j < i,\ m[j] \geq m_k[j]\},\\ \mathrm{sim}(k,i) &= \max_{1\leq j\leq k}\{j \mid \exists m \in I(k,i) \text{ such that } \forall l,\ l < j \text{ implies } m[l] \leq m_k[l]\}. \end{aligned}$$

Using Theorem 2, we are able to prove a generalization of a well-known characterization of Horn sets. A related characterization in a different setting can be found in [9].

Proposition 3. *A set of vectors M over a finite totally ordered domain is closed under conjunction if and only if there exists a Horn constraint φ satisfying the identity $\mathrm{Sol}(\varphi) = M$.*

If we interchange conjunctions with disjunctions of models, as well as positive and negative literals throughout Section 4, we obtain identical results for dual Horn constraints.

Theorem 4. *A set of vectors $M \subseteq D^\ell$ over a finite ordered domain D is closed under disjunction if and only if there exists a dual Horn constraint φ satisfying the identity $M = \mathrm{Sol}(\varphi)$. Given M closed under disjunction, the dual Horn constraint φ contains at most $3\,|M|\,\ell$ clauses and its length is $O(|M|\,\ell^2\log|D|)$. It can be constructed in time $O(|M|\,\ell(|M|+\ell)\log|D|)$ and space $O(|M|\,\ell\log|D|)$.*

5 Bijunctive Constraints

Bijunctive clauses and formulas present another frequently studied subclass of propositional formulas. We investigate in this section the description problem for a generalization of bijunctive formulas to ordered finite domains, namely for sets of vectors closed under the median operation.

Problem: DESCRIPTION[BIJUNCTIVE]
Input: A finite set of vectors $M \subseteq D^\ell$, closed under median, over a finite totally ordered domain D.
Output: A bijunctive constraint φ over D such that $\mathrm{Sol}(\varphi) = M$.

The general construction in Section 3 does not guarantee that the final constraint is bijunctive whenever the set M is closed under median. Therefore we add a post-processing step that transforms the constraint into a bijunctive one. Let $\varphi(M)$ be the constraint produced by the method of Section 3 and let c be a clause from $\varphi(M)$. We construct a bijunctive restriction $b(\varphi)$ by removing appropriate literals from φ such that no more than two literals remain in each clause. Since φ is a conjunctive normal form, any model of $b(\varphi)$ is still a model of φ. The converse does not hold in general. However, if $\mathrm{Sol}(\varphi)$ is closed under median, the method presented below preserves the models, i.e., every model of φ remains a model of $b(\varphi)$. In the proof we need a simple lemma.

Lemma 5. *The model* $\mathrm{med}(m_1, m_2, m_3)$ *satisfies a literal* l *if and only if at least two of the models* m_1*,* m_2*, and* m_3 *satisfy* l*.*

We say that a literal l is *essential* for a clause c if there is a model $m \in M$ that satisfies l, but no other literal in c; we also say that m is a *justification* for l. Obviously, we may remove non-essential literals from c without losing models. It remains to show that no clause from φ contains more than two essential literals.

To derive a contradiction, suppose that c is a clause from φ containing at least three essential literals, say l_1, l_2, and l_3. Let m_1, m_2, and m_3 be their justifications, i.e., for each i we have $m_i \models l_i$ and m_i does not satisfy any other literal in c. According to Lemma 5, in this case the model $\mathrm{med}(m_1, m_2, m_3)$ satisfies no literal at all. Hence $\mathrm{med}(m_1, m_2, m_3)$ satisfies neither c nor φ, which contradicts the assumption that $\mathrm{Sol}(\varphi)$ is closed under median.

The preceding discussion suggests applying the following algorithm to every clause c of φ. For every literal l in $c = c' \vee l$, check whether the remaining clause c' is still satisfied by all models in M. If yes, the literal is not essential and can be removed. Otherwise it is one of the two literals in the final bijunctive clause $b(c)$.

Proposition 6. *Given a finite set of vectors* $M \subseteq D^\ell$ *over a finite totally ordered domain* D*, closed under median, and a constraint* φ *in conjunctive normal form such that* $M = \mathrm{Sol}(\varphi)$*, an equivalent bijunctive constraint* $b(M)$ *with the same number of clauses can be computed in time* $O(d\,|M|\,\ell \log |D|)$ *and space* $O(|M|\,\ell \log |D|)$*, where* d *is the number of clauses in* φ*.*

Theorem 7. *For each set of vectors* $M \subseteq D^\ell$ *over a finite ordered domain* D *that is closed under median, there exists a bijunctive constraint* φ *such that* $M = \mathrm{Sol}(\varphi)$*. Its length is* $O(|M|\,\ell(\log \ell + \log |D|))$ *and it contains at most* $3\,|M|\,\ell$ *clauses. The algorithm constructing* φ *runs in time* $O(|M|^2\,\ell^2 \log |D|)$ *and space* $O(|M|\,\ell \log |D|)$*.*

Proposition 8. *A set of vectors* M *over a finite totally ordered domain is closed under median if and only if there exists a bijunctive constraint* φ *satisfying the identity* $M = \mathrm{Sol}(\varphi)$*.*

We wish to point out that reducing the length of clauses also allows us to compute a *Horn* constraint whenever the set M is closed under conjunction. In this case a *minimal* clause in a constraint φ with $M = \mathrm{Sol}(\varphi)$ is always Horn.

Contrary to the Horn case, the simplest known algorithm for the bijunctive description problem does not seem to lift well from the Boolean to the finite domain. Dechter and Pearl [7] showed that in the Boolean case this problem can be solved in time $O(|M|\,\ell^2)$, which is better than our result even when ignoring the unavoidable factor $\log|D|$. Their algorithm generates first all the $O(\ell^2)$ bijunctive clauses built from the variables of the formula, followed by an elimination of those falsified by a vector from M, where the bijunctive formula is the conjunction of the retained clauses. However, there are $O(\ell^2\,|D|^2)$ bijunctive clauses for a finite domain D yielding an algorithm with time complexity $O(|M|\,\ell^2\,|D|^2)$, which is exponential in the size $O(\log|D|)$ of the domain elements.

Another idea, not applicable efficiently in the finite domain case, is that of projecting M onto each pair of variables, then computing a bijunctive constraint for each projection. This requires time $O(|M|\,\ell^2)$ in the Boolean case, since we only need to compute a CNF for each projection. A CNF for a projection is always bijunctive, thus only the general, efficient algorithm of Theorem 1 has to be used. However, in the finite domain case, computing a constraint with the algorithm of Theorem 1 does not necessarily yield a bijunctive one. Each clause can contain up to four literals, a positive and a negative one for each variable. Thus we need to use an algorithm for computing a *bijunctive* CNF, like that of Theorem 7, yielding an overall time complexity of $O(|M|^2\,\ell^2\log|D|)$.

6 Affine Constraints

Recall that a set of vectors over D is *affine* if it is a Cartesian product of affine spaces (one for each prime factor of n), i.e., of vector spaces translated by some vector. For sake of completeness this section summarizes some results from the theory of finite fields relevant for the analysis of affine constraints. An interested reader can find more information e.g. in the monograph [18].

The main question in the case of affine constraints and affine sets is whether the cardinality n of the domain is prime or not. According to this, we distinguish three cases: (1) n is prime, which will be studied in full detail; (2) n is a product of prime factors, which will reduced by means of the Chinese Remainder Theorem to the previous case; and (3) n is a prime power, which will be reduced by means of Hensel lifting to the first case.

6.1 n is prime

If n is prime then the working domain D is the finite field $\mathbb{Z}_n$, sometimes denoted also $\mathbb{F}_n$ or $\mathbb{Z}/n\mathbb{Z}$. Recall first some necessary results from linear algebra. Let $\varphi(x)$ be an affine constraint equivalent to the affine system $Ax = b$ over the finite field $\mathbb{Z}_n$, where A is a $p \times q$ matrix over $\mathbb{Z}_n$. If the system $Ax = b$ is consistent, i.e., it does not imply an equation $a_1 = a_2$ for two different values

$a_1 \neq a_2$ from $\mathbb{Z}_n$, and of full row rank, then it has n^{q-p} solutions. The solutions of the system $Ax = b$ form an affine space, i.e., a vector space translated by a vector. The set of solutions M of the system $Ax = b$ can be written as a direct sum of the solutions M^* of the homogeneous system $Ax = \mathbf{0}$ and a particular solution m of the system $Ax = b$. The set of solutions M^* form a vector space and the particular solution m is the translating vector.

This section investigates the description problem for sets of vectors representing an affine space.

Problem: DESCRIPTION[AFFINE], n PRIME
Input: A finite set of vectors $M \subseteq D^\ell$, representing an affine space over a finite totally ordered domain D.
Output: An affine constraint φ over D such that $\mathrm{Sol}(\varphi) = M$.

The first point to check in an affine description problem is whether the cardinality of a set of vectors M over $\mathbb{Z}_n$ is a power of n, otherwise M cannot be an affine space. Then we choose an arbitrary vector $m^* \in M$ and form the set $M^* = \{m - m^* \mid m \in M\}$. It is clear that M is an affine space if and only if M^* is a vector space. If the cardinality of M^* is equal to n^q for some q, then there must exist a homogeneous linear system $(I\ B)(u\ v) = \mathbf{0}$ over $\mathbb{Z}_n$, where I is an $(\ell - q) \times (\ell - q)$ identity matrix and B is a $(\ell - q) \times q$ matrix of full row rank over $\mathbb{Z}_n$. We will concentrate on the construction of the matrix B.

The i-th row of the system $(I\ B)(u\ v) = \mathbf{0}$ is $u_i + b_i^1 v_1 + \cdots + b_i^q v_q = 0$, or $u_i + b_i v = 0$, where b_i is the i-th row of the matrix B. Let m be a vector from the set M^*. We substitute $m[i]$ for u_i and $m[\ell - q + j]$ for v_j for each $j = 1, \ldots, q$. This implies the equation $m[i] + b_i^1 m[\ell - q + 1] + \cdots + b_i^q m[\ell] = 0$, or $(e_i\ b_i)m = 0$, for each $m \in M^*$, where e_i is the corresponding unit vector. This means that we construct the homogeneous system $S_i \colon M^*(e_i\ b_i) = \mathbf{0}$, where M^* is the matrix whose rows are the vectors of the set M^*. The system S_i is an inhomogeneous system over $\mathbb{Z}_n$ with q variables b_i, since the dot product of the unit vector e_i with each vector $m \in M$ produces the constant $m[i]$. If M^* is a vector space, then the system S_i has exactly one solution, which constitutes the i-th row of the matrix B. This is because the vector space M^* of cardinality n^q has the dimension q, i.e., each basis of M^* contains exactly q linearly independent vectors. If S_i has no solution, then M^* is not a vector space. The system S_i cannot have more than one solution, which follows from the cardinality of the set M^* and the construction of the system S_i.

Once all the systems S_i have been solved, we have determined the coefficients of the matrix B in the homogeneous system $(I\ B)(u\ v) = \mathbf{0}$. To determine the inhomogeneous system $(I\ B)(u\ v) = b$ that describes the original set of vectors M, we substitute the vector m^* for the variables $(u\ v)$ and derive the values of the vector b. This implies the following result.

Theorem 9. *If $M \subseteq \mathbb{Z}_n^\ell$ is a set of vectors representing an affine space then there exists an affine system $Ax = b$ with $\ell - \log|M| / \log|D|$ rows over $\mathbb{Z}_n$, such that $\mathrm{Sol}(Ax = b) = M$, that can be computed in time $O(|M|^{mx}(\ell \log|D| +$*

$\log|M|)\log|D|)$ *and space* $O((|M|+\ell)\ell\log|D| - \ell\log|M|)$*, where* mx *is the exponent in the asymptotic complexity for matrix multiplication.*

Following the preceding discussion, there exists an easy way to determine whether a set of vectors M over $\mathbb{Z}_n$ is an affine space. Since we work in a ring, we can also use the subtraction operation over $\mathbb{Z}_n$, making the characterization more compact.

Proposition 10. *A set of vectors* $M \subseteq \mathbb{Z}_n^\ell$ *is an affine space if and only if it is closed under the affine operation* $\mathrm{aff}(x,y,z) = x - y + z \pmod n$*, i.e., for any choice of three not necessarily distinct vectors* $m, m', m'' \in M$ *the vector* $m - m' + m''$ *also belongs to* M*.*

6.2 n is not prime

If n is not prime, then it can be written as a product of relatively prime factors $n = n_1 \cdots n_q$. We assume that this factorization is *a priori* known, since it does not make sense to factorize n every time we need to solve an affine problem over $\mathbb{Z}_n$. The affine description problem is then formulated as follows.

Problem: DESCRIPTION[AFFINE], n COMPOSED
Input: A finite set of vectors $M \subseteq D^\ell$, representing a Cartesian product of affine spaces over a finite totally ordered domain D.
Output: An affine constraint φ over D such that $\mathrm{Sol}(\varphi) = M$.

A lot of results can be reused from the previous case, when n is prime, but we need to use the Chinese Remainder Theorem to solve the affine systems.

Theorem 11 (Chinese Remainder Theorem). *Let* $n_1, \ldots, n_q$ *be pairwise relatively prime and* $n = n_1 \cdots n_q$*. Consider the mapping* $f\colon a \leftrightarrow (a_1, \ldots, a_q)$*, where* $a \in \mathbb{Z}_n$*,* $a_i \in \mathbb{Z}_{n_i}$*, and* $a_i = a \mod n_i$*, for each* $i \in \{1, \ldots, q\}$*. Then the mapping* f *is a bijection between* $\mathbb{Z}_n$ *and the Cartesian product* $\mathbb{Z}_{n_1} \times \cdots \times \mathbb{Z}_{n_q}$*.*

The Chinese Remainder Theorem says that operations performed on the elements of $\mathbb{Z}_n$ can be equivalently performed on the corresponding q-tuples by performing the operations independently in each coordinate position in the appropriate system. Instead of working with an affine system of equations $Ax = b$ over $\mathbb{Z}_n$, we work with q affine systems $Ax = b$ over $\mathbb{Z}_{n_k}$. The factorization $n = n_1 \cdots n_q$ implies that $q \le \log n$, since the inequality $n_k \ge 2$ holds for each k. This means that we have only a logarithmic number of subsystems to consider. Since the length of n is $\log n$, this means that we have only a polynomial number (with respect to the length of input) of systems $Ax = b$ over $\mathbb{Z}_{n_i}$ to consider.

The constraint description algorithm uses the Chinese Remainder Theorem in the other direction. In practice, we compute first the coefficients of the matrix B_k, as in Section 6.1, modulo n_k for each $k = 1, \ldots, q$. By means of the mapping f in the Chinese Remainder Theorem we determine from $B_1, \ldots, B_q$ the matrix B of the homogeneous system $(I\ B)(u\ v) = \mathbf{0}$. The vector b of the final inhomogeneous system $(I\ B)(u\ v) = b$ is determined as in Section 6.1. An application of the

Chinese Remainder Theorem requires $O(\mathrm{im}(\log |D|) \log\log |D|)$ operations [18, Theorem 10.25], where $\mathrm{im}(r)$ denotes the time for multiplication of two integers of length r.

Theorem 12. *If $M \subseteq \mathbb{Z}_n^\ell$ is a set of vectors representing a Cartesian product of affine spaces then there exists a system $Ax = b$ over $\mathbb{Z}_n$, where $\mathrm{Sol}(Ax = b) = M$.*

A factor n_i in the previous factorization $n = n_1 \cdots n_q$ need not be prime, it can also be a prime power. Therefore we need to consider a third case.

6.3 n is a prime power

If $n = p^q$ for some prime p and an integer exponent $q > 1$, then we use Hensel lifting to solve our problem. Since the Hensel lifting is by far beyond the scope of this paper, we only state the main result without any particular presentation of the lifting method. An interesting reader is strongly encouraged to consult the part on Hensel lifting in the monograph [18]. We assume that the power p^q is known to us in binary notation, therefore its length is $O(q \log p)$. We compute first the system $(I\ B)(u\ v) = b \pmod p$, as in Section 6.1, followed by a Hensel lifting to the system $A'x = b' \pmod{p^{2^{\lceil \log_2 k \rceil}}}$. Each coefficient $B(i,j)$ and b_i of the matrix B and the vector b, respectively, is lifted separately, similarly to the application of the Chinese Remainder Theorem in Section 6.2. Finally, we cut the result down to modulo $m = p^q$. Usually, Hensel lifting is presented for polynomials over a ring R. To adapt our approach to the usual presentation, we use the same trick as that applied for cyclic codes: a number $a = a_d a_{d-1} \cdots a_1 a_0$ is interpreted as a polynomial $a_d X^d + a_{d-1} X^{d-1} + \cdots + a_1 X + a_0$, where X is a formal variable. The description problem for affine constraints is formulated as follows.

Problem: DESCRIPTION[AFFINE], n PRIME POWER
Input: A finite set of vectors $M \subseteq D^\ell$, representing a power of an affine space over a finite totally ordered domain D.
Output: An affine constraint φ over D such that $\mathrm{Sol}(\varphi) = M$.

Each Hensel step requires $O(\mathrm{im}(\log |D|)\, \mathrm{im}(\log p))$ operations, where $\mathrm{im}(r)$ is the time for multiplying two integers of length r. There are $O(\log q)$ iterations of the Hensel step. Hence, there exists a polynomial-time algorithm to compute the system $A'x = b' \pmod{p^q}$ from the system $(I\ B)(u\ v) = b \pmod p$.

Theorem 13. *If $M \subseteq \mathbb{Z}_n^\ell$ is a set of vectors representing a power of an affine space then there exists an affine system $Ax = b$ over $\mathbb{Z}_n$, where $\mathrm{Sol}(Ax = b) = M$.*

7 Changing the Literals

If we change the underlying notion of literals, using $x = d$ and $x \neq d$ as basic building blocks, the situation changes drastically. Former positive literal $x \geq d$ becomes a shorthand for the disjunction $(x = d) \vee (x = d+1) \vee \cdots \vee (x = n-1)$, whereas the former negative literal $x \leq d$ now represents the disjunction

$(x = 0) \vee (x = 1) \vee \cdots \vee (x = d)$. Even if we compress literals containing the same variable into a bit vector, the new representation still needs n bits, i.e., its size is $O(n)$. Compared to the former literals of size $O(\log n)$, this amounts to an exponential blow-up. As an immediate consequence the algorithms given in the preceding sections become exponential, since we have to replace literals like $x_i < m_k[i]$, $x_i > m_k[i]$, and $x_i < m_{k+1}[i]$ by disjunctions of equalities.

The satisfiability problem for constraints in CNF over finite totally ordered domains with basic operators $\leq$ and $\geq$ is defined similarly to Boolean satisfiability. The complexity of these problems was studied for fixed domain cardinalities, from the standpoint of many-valued logics, by Hähnle *et al.* [3, 10]. The NP-completeness proof for Boolean satisfiability generalizes uniformly to finite ordered domains. Hähnle *et al.* [3, 10] proved that the satisfiability problems restricted to Horn, dual Horn, and bijunctive constraints, are decidable in polynomial time for a fixed domain cardinality. The tractability of the affine restriction is a consequence of Section 6.

The satisfiability of constraints in conjunctive normal form is also affected when switching to $=$ and $\neq$ as basic operators. While the satisfiability problem for general constraints remains NP-complete, the restrictions to Horn, dual Horn, and bijunctive constraints change from polynomially solvable to NP-complete for $|D| \geq 3$. This can be shown by encoding for example the graph problem of k-COLORING [1, 5]. When we use the Horn and bijunctive clause $(u \neq d \vee v \neq d)$, we can express by $C(u, v) = (u \neq 0 \vee v \neq 0) \wedge \cdots \wedge (u \neq k-1 \vee v \neq k-1)$ that the adjacent vertices of the edge (u, v) are "colored" by different "colors". On the other hand, Beckert *et al.* [2] proved that bijunctive constraints restricted to positive literals can be solved in linear time.

8 Concluding Remarks

The studied constraint description problems constitute a generalization of the Boolean structure identification problems, studied by Dechter and Pearl [7], with more efficient algorithms as a byproduct. Our paper presents a complement to the work of Hähnle *et al.* [10] on the complexity of the satisfiability problems in many-valued logics. It also completes the study of tractable constraints [5, 14, 15] by Jeavons and his group.

We have constructed efficient polynomial-time algorithms for constraint description problems over a finite totally ordered domain, where the produced constraint is in conjunctive normal form. If the original set of vectors is closed under the operation of conjunction, disjunction, or median, we have presented specific algorithms that produce a Horn, a dual Horn, or a bijunctive constraint, respectively. In all three cases, the constraint contains at most $3\,|M|\,\ell$ clauses. It is interesting to note that the produced algorithms are compatible, with respect to asymptotic complexity, with known algorithms for the Boolean case presented in [11, 19]. This means that the restriction of the new algorithms presented in our paper to domains D with cardinality $|D| = 2$ produces the aforementioned algorithms for the Boolean case. However, the presented algorithms are not just

straightforward extensions of the previous ones for the Boolean case, but they required the development of new methods.

It would be interesting to know if more efficient algorithms exist or whether our algorithms are asymptotically optimal. Certainly a more involved lower bound analysis is necessary to answer this open question. A possible extension of our work would be a generalization of our algorithms to partially ordered domains and to domains with a different structure, like lattices.

References

1. C. Ansótegui and F. Manyà. New logical and complexity results for signed-SAT. In *Proc. 33rd ISMVL 2003, Tokyo (Japan)*, pages 181–187. 2003.
2. B. Beckert, R. Hähnle, and F. Manyà. The 2-SAT problem of regular signed CNF formulas. In *Proc. 30th ISMVL, Portland (OR, USA)*, pages 331–336. 2000.
3. R. Béjar, R. Hähnle, and F. Manyà. A modular reduction of regular logic to classical logic. In *Proc. 31st ISMVL, Warsaw (Poland)*, pages 221–226. 2001.
4. A. A. Bulatov. A dichotomy theorem for constraints on a three-element set. In *Proc. 43rd FOCS, Vancouver (BC, Canada)*, pages 649–658, 2002.
5. M. C. Cooper, D. A. Cohen, and P. Jeavons. Characterising tractable constraints. *Artificial Intelligence*, 65(2):347–361, 1994.
6. N. Creignou, S. Khanna, and M. Sudan. *Complexity Classifications of Boolean Constraint Satisfaction Problems*. SIAM Monographs on Discrete Mathematics and Applications. SIAM, Philadelphia (PA), 2001.
7. R. Dechter and J. Pearl. Structure identification in relational data. *Artificial Intelligence*, 58(1-3):237–270, 1992.
8. T. Feder and M. Y. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: a study through Datalog and group theory. *SIAM Journal on Computing*, 28(1):57–104, 1998.
9. R. Hähnle. Exploiting data dependencies in many-valued logics. *Journal of Applied Non-Classical Logics*, 6(1), 1996.
10. R. Hähnle. Complexity of many-valued logics. In *Proc. 31st ISMVL, Warsaw (Poland)*, pages 137–148. 2001.
11. J.-J. Hébrard and B. Zanuttini. An efficient algorithm for Horn description. *Information Processing Letters*, 88(4):177–182, 2003.
12. P. Hell and J. Nešetřil. On the complexity of H-coloring. *Journal of Combinatorial Theory, Series B*, 48:92–110, 1990.
13. P. Jeavons. On the algebraic structure of combinatorial problems. *Theoretical Computer Science*, 200(1-2):185–204, 1998.
14. P. Jeavons, D. Cohen, and M. Gyssens. Closure properties of constraints. *Journal of the Association for Computing Machinery*, 44(4):527–548, 1997.
15. P. Jeavons and M. C. Cooper. Tractable constraints on ordered domains. *Artificial Intelligence*, 79(2):327–339, 1995.
16. P. G. Kolaitis and M. Y. Vardi. Conjunctive-query containment and constraint satisfaction. *Journal of Computer and System Science*, 61(2):302–332, 2000.
17. T. J. Schaefer. The complexity of satisfiability problems. In *Proc. 10th STOC, San Diego (CA, USA)*, pages 216–226, 1978.
18. J. von zur Gathen and J. Gerhard. *Modern Computer Algebra*. Cambridge University Press, 1999.
19. B. Zanuttini and J.-J. Hébrard. A unified framework for structure identification. *Information Processing Letters*, 81(6):335–339, 2002.

Appendix

Proposition 3. *A set of vectors M over a finite totally ordered domain is closed under conjunction if and only if there exists a Horn constraint φ satisfying the identity* $\mathrm{Sol}(\varphi) = M$.

Proof. Theorem 2 shows that there exists a Horn constraint φ describing a set of vectors M if M is closed under conjunction. It remains to show the converse, namely that the set $\mathrm{Sol}(\varphi)$ is closed under conjunction if φ is a Horn constraint. We need to show that for any two models m and m' of φ, their conjunction $m \wedge m'$ is also a model of φ. A model satisfies a Horn constraint φ if and only if it satisfies every clause of φ. Therefore we need to show for each clause c that $m \wedge m'$ satisfies c whenever both m and m' satisfy c. We distinguish two cases:

- Clause c contains a positive literal $x \geq d$ that is satisfied by both m and m'. Then we have $m(x) \geq d$ and $m'(x) \geq d$, which implies $(m \wedge m')(x) = \min(m(x), m'(x)) \geq d$. This proves that $m \wedge m'$ satisfies the clause c.
- Clause c contains no positive literal satisfied by both m and m'. Then at least one model, say m, must satisfy a negative literal $x \leq d$ in c, i.e., $m(x) \leq d$. We obtain $(m \wedge m')(x) = \min(m(x), m'(x)) \leq m(x) \leq d$. Hence also $m \wedge m'$ satisfies c.

This case analysis terminates the proof. □

Lemma 5. *The model* $\mathrm{med}(m_1, m_2, m_3)$ *satisfies a literal l if and only if at least two of the models m_1, m_2, and m_3 satisfy l.*

Proof. If the literal l is satisfied by at least two models, say m_1 and m_2, then it is satisfied also by the disjuncts $m_1 \vee m_2$, $m_2 \vee m_3$, and $m_3 \vee m_1$. Hence, l is also satisfied by $\mathrm{med}(m_1, m_2, m_3)$.

Conversely, if l is satisfied only by one model, say m_1, or if l is not satisfied by any of the three models, then it is falsified by the disjunct $m_2 \vee m_3$. Hence, it is *not* satisfied by $\mathrm{med}(m_1, m_2, m_3)$. □

Proposition 6. *Given a finite set of vectors $M \subseteq D^\ell$ over a finite totally ordered domain D, closed under median, and a constraint φ in conjunctive normal form such that $M = \mathrm{Sol}(\varphi)$, an equivalent bijunctive constraint $b(M)$ with the same number of clauses can be computed in time $O(d\,|M|\,\ell \log |D|)$ and space $O(|M|\,\ell \log |D|)$, where d is the number of clauses in φ.*

Proof. The algorithm requires time $O(|M|\,\ell \log |D|)$ per literal, since we have to check each of at most $2\ell - 1$ remaining literals against each model, with each comparison taking time $O(\log |D|)$. The whole clause c contains at most 2ℓ literals, therefore we need time $O(|M|\,\ell^2 \log |D|)$. However, we can do better by using an idea described in [19] for the Boolean case. We scan the set of models only once per clause and record in a table the literals satisfied by each model. We select the essential literals in the second step. This algorithm requires only time $O(|M|\,\ell \log |D|)$ per clause. For details see [19, Section 4]. □

Theorem 7. *For each set of vectors $M \subseteq D^\ell$ over a finite ordered domain D that is closed under median, there exists a bijunctive constraint φ such that $M = \mathrm{Sol}(\varphi)$. Its length is $O(|M|\,\ell(\log \ell + \log |D|))$ and it contains at most $3\,|M|\,\ell$ clauses. The algorithm constructing φ runs in time $O(|M|^2\,\ell^2 \log |D|)$ and space $O(|M|\,\ell \log |D|)$.*

Proof. The main part of the result follows from Theorem 1 and Proposition 6. Concerning the length of φ, note that each clause contains at most two literals. Each literal consists of a variable and one domain element. Hence we can represent a literal by the index of its variable and the domain value, both written in binary. Therefore the length of a literal and subsequently of each bijunctive clause is $O(\log \ell + \log |D|)$. Since there are at most $3\,|M|\,\ell$ clauses, the length of the bijunctive constraint follows. □

Proposition 8. *A set of vectors M over a finite totally ordered domain is closed under median if and only if there exists a bijunctive constraint φ satisfying the identity $M = \mathrm{Sol}(\varphi)$.*

Proof. Theorem 7 shows that there exists a bijunctive constraint for every set of vectors M closed under median. It remains to show the converse, namely that $\mathrm{Sol}(\varphi)$ is closed under median if φ is a bijunctive constraint. Since φ is a conjunction of clauses, it is sufficient to show the closure property for a bijunctive clause $c = l \vee l'$. Let m_1, m_2, and m_3 be three models of φ. From the pigeonhole principle follows that one of the two literals l or l' of c is satisfied by at least two models. Hence, by Lemma 5, the literal and therefore also the clause c is satisfied by $\mathrm{med}(m_1, m_2, m_3)$. □

Proposition 10. *A set of vectors $M \subseteq \mathbb{Z}_n^\ell$ is an affine space if and only if it is closed under the affine operation $x - y + z \pmod n$, i.e., for any choice of three not necessarily distinct vectors $m, m', m'' \in M$ the vector $m - m' + m''$ also belongs to M.*

Proof. If M is an affine space then for each vector $m^* \in M$, the set $M^* = \{m - m^* \mid m \in M\}$ is a vector space. Let $m_1, m_2, m_3 \in M$ be three vectors from the affine space. Then there exist corresponding vectors $m_i^* = m_i - m^*$ for each $i = 1, 2, 3$ in the vector space M^*. A vector space is closed under linear combination of vectors. Therefore also the vector $m_1^* + (n-1)m_2^* + m_3^*$, which is equal to $m_1^* - m_2^* + m_3^*$, belongs to M^*, which implies that the vector $m_1^* - m_2^* + m_3^* + m^*$ belongs to the affine space M. Since the equality

$$\begin{aligned} m_1^* - m_2^* + m_3^* + m^* &= m_1^* - m_2^* + m_3^* + m^* + m^* - m^* \\ &= (m_1^* + m^*) - (m_2^* + m^*) + (m_3^* + m^*) \\ &= m_1 - m_2 + m_3 \end{aligned}$$

holds, we get that the affine space M is closed under the affine operation.

Suppose that M is closed under the affine operation $\mathrm{aff}(x, y, z)$, i.e., for all $m_1, m_2, m_3 \in M$ we have $m_1 - m_2 + m_3 \in M$. Fix a vector $m^* \in M$ and subtract it from all vectors in M, constructing the set $M^* = \{m - m^* \mid m \in M\}$. The closure under the affine operator implies that for each choice of two not necessarily distinct vectors $m_1, m_2 \in M$, also the vector $m_1 - m_2 + m^*$ belongs to M, and therefore $m_1 - m_2$ belongs to M^*. In addition to the fixed vector $m^* \in M$, choose the vectors $m_a, m_b, m_c \in M$. From the construction of the set M^* we know that both vectors $m_a^* = m_a - m^*$ and $m_b^* = m_b - m^*$ belong to M^*. Construct by means of the affine operator the vectors $m_1 = m_a - m^* + m_c$ and $m_2 = m^* - m_b + m_c$. From the closure under the affine operator follows that both vectors m_1 and m_2 belong to the set M. Hence, the vector

$$\begin{aligned} m_1 - m_2 &= (m_a - m^* + m_c) - (m^* - m_b + m_c) \\ &= (m_a - m^*) + (m_b - m^*) \\ &= m_a^* + m_b^* \end{aligned}$$

belongs to the set M^*. This means that for any choice of not necessarily distinct vectors $m_a^*, m_b^* \in M^*$ we have that $m_a^* + m_b^* \in M^*$. By induction we get that $\lambda_1 m_1^* + \cdots + \lambda_k m_k^* \in M^*$ for all scalars $\lambda_1, \ldots, \lambda_k \in D$ and all vectors $m_1^*, \ldots, m_k^* \in M^*$. This is just the same as saying that the set M^* is closed under linear combination. In addition, the all-zero vector $\mathbf{0} = m^* - m^*$ belongs to M^*, therefore we proved that M^* is a vector space. Since M is the set of vectors M^* translated by the vector m^*, the set M constitutes an affine space. $\square$