Complexity of Clausal Constraints Over Chains*

Nadia Creignou†
LIF (CNRS, UMR 6166)
Univ. de la Méditerranée
13288 Marseille cedex 9, France

Miki Hermann‡
LIX (CNRS, UMR 7161)
École Polytechnique
91128 Palaiseau cedex, France

Andrei Krokhin§
Department of Computer Science
University of Durham
Durham, DH1 3LE, United Kingdom

Gernot Salzer¶
Technische Universität Wien
Favoritenstraße 9-11
A-1040 Wien, Austria

Abstract

We investigate the complexity of the satisfiability problem of constraints over finite totally ordered domains. In our context, a clausal constraint is a disjunction of inequalities of the form $x \geq d$ and $x \leq d$. We classify the complexity of constraints based on clausal patterns. A pattern abstracts away from variables and contains only information about the domain elements and the type of inequalities occurring in a constraint. Every finite set of patterns gives rise to a (clausal) constraint satisfaction problem in which all constraints in instances must have an allowed pattern. We prove that every such problem is either polynomially decidable or NP-complete, and give a polynomial-time algorithm for recognizing the tractable cases. Some of these tractable cases are new and have not been previously identified in the literature.

1 Introduction

Research in complexity of constraint satisfaction problems (CSP) gained a considerable interest in the recent years. The first complete classification by means of a dichotomy theorem for Boolean CSP of Schaefer [Sch78] has been followed in the last decade by an intensive effort to extend his result to larger domains. Feder and Vardi [FV98] conjectured that a dichotomy theorem holds for every finite domain. This conjecture has been partially confirmed in 2002 when Bulatov [Bul02] proved a dichotomy theorem for the 3-element domain. The problem remains open for domains of higher cardinality. Confronted with the difficulty of the main goal, researchers started to investigate CSP problems with additional structure, like list constraints or conservative constraints [Bul03]. The effort has been pursued along several lines: one of them applies methods from universal algebra [BJK05, Bul02, Bul03], the other one is oriented towards graph theoretic methods [FV98, HN04], and there is also a finite model theory approach [Dal02, FV98, KV00].

*The work has been supported by ÉGIDE 06606ZF and ÖAD Amadeus 18/2004.
†email: Nadia.Creignou@lif.univ-mrs.fr
‡email: Miki.Hermann@lix.polytechnique.fr
§email: Andrei.Krokhin@durham.ac.uk
¶email: salzer@logic.at

Dichotomy results are important in computational complexity. Ladner proved in [Lad75] that there exists an infinite hierarchy between the class P of polynomial-time decidable problems and NP-complete problems, provided that P $\neq$ NP. On the other hand, a dichotomy theorem states that a considered constraint satisfaction problem is either polynomial-time decidable or NP-complete, depending on a parameter usually presented in the form of a finite set of relations.. This means that dichotomy results are not at all obvious and should constitute an exception under the hypothesis that P $\neq$ NP.

In this paper we consider constraint satisfaction problems inspired by many-valued logic. In fact, we take for basis of our research the regular signed logic [Häh01], where the underlying finite domain is totally ordered, i.e., is a chain. This logic provides us with the concept of literal, clause, and formula, which extend naturally from the Boolean domain. Moreover, the well-known polynomial-time decidable satisfiability cases, namely Horn, dual Horn, bijunctive, 0- and 1-valid preserve their good complexity properties. The cornerstone of our approach is the concept of clausal pattern, which is an abstraction of all clauses of certain type. These patterns correspond, roughly speaking, to constraints in the algebraic approach. Finite sets of patterns constitute clausal languages upon which we construct formulas, whose satisfiability are at the heart of our CSP problems. Naturally, these CSP problems are parametrized by clausal languages. Different clausal languages lead to CSP problems of different complexity. First, we get rid of the redundant values in the domain, and thus concentrate on clausal languages closed under taking subpatterns and containing all unary clauses. Next, to be able to compare clausal languages and subsequently the complexity of the induced CSP problems, we build 3-saturations by means of resolution and subpatterns. Roughly speaking, the 3-saturation of a clausal language L gives all patterns of length at most 3 that can be "implemented" by L, thus measuring the expressive power of L.

We first derive the exact condition for NP-complete CSP problems, followed by a careful and exhaustive analysis on the patterns occurring in the saturation, which provide us with the polynomial-time decidable cases. We obtain a dichotomy theorem for any cardinality of the finite totally ordered domain, where the previously known four polynomial-time decidable cases keep their complexity. Moreover, we discover new polynomial-time decidable cases, which are absent from Boolean CSP problems. We present an algorithm computing a satisfying assignment of a formula, if it exists, by a unified approach for all polynomial-time decidable cases. Hence, we obtain a complete classification of many-valued logics-based CSP problems over finite totally ordered domains.

2 Preliminaries

Let D be a finite chain, say $D = \{0, 1, \ldots, n-1\}$ with the total order $0 < 1 < \cdots < n-1$, called a *domain*, 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. A *clause* is a disjunction of literals. As usually, an *interval* $[a, b]$ denotes the subset of values $\{x \in D \mid a \leq x \leq b\}$.

A *clausal pattern*, or *pattern* for short, is a multiset[1] of the form

$$P \quad = \quad (+a_1, \ldots, +a_p, -b_1, \ldots, -b_q),$$

where $p, q \in \mathbb{N}$ and a_i, b_i are values from the domain D, for all i. The $+a_i$'s are called positive literals, the $-b_i$'s negative literals. The sum $p + q$, also denoted by $|P|$, is the *length* of the

[1] A multiset (also called bag) is a set with repetitions.

pattern. If P and Q are patterns and all literals of P occur in Q then P is called a *subpattern* of Q. A *clausal language* L is a finite set of clausal patterns, with arities not necessarily equal. We denote by P^+ and P^- the positive and negative parts, respectively, of the pattern P, i.e., $P^+ = (+a_1, \ldots, +a_p)$ and $P^- = (-b_1, \ldots, -b_q)$. We denote by $\min(P^+) = \min\{a_1, \ldots, a_p\}$ and $\max(P^-) = \max\{b_1, \ldots, b_q\}$ the *minimum* and *maximum* values of the parts P^+ and P^-, respectively. We denote by $\mathrm{pos}_\ell(L) = \{P \in L \mid P = P^+, |P| = \ell\}$ and $\mathrm{neg}_\ell(L) = \{N \in L \mid N = N^-, |N| = \ell\}$ the set of all positive and negative patterns of length ℓ in L, respectively.

Given a clausal language L and a set of variables V, an *L-clause* is a pair $(P, \vec{x})$, where $P \in L$ is a pattern and $\vec{x}$ is a vector of not necessarily distinct variables from V, such that $|P| = |\vec{x}|$. A pair $(P, \vec{x})$ with a pattern $P = (+a_1, \ldots, +a_p, \ldots, -b_1, \ldots, -b_q)$ and variables $\vec{x} = (x_1, \ldots, x_{p+q})$ represents the clause $c_P = (x_1 \geq a_1 \vee \cdots \vee x_p \geq a_p \vee x_{p+1} \leq b_1 \vee \cdots \vee x_{p+q} \leq b_q)$. We will use the more conventional notation $P(\vec{x})$ instead of $(P, \vec{x})$. An *L-formula*, or *formula* for short, is a conjunction of a finite number of L-clauses. We denote by $\varphi(x_1, \ldots, x_k)$ that the variables $x_1, \ldots, x_k$ occur in φ, fixing also implicitly their sequence.

An *assignment* is a mapping $I \colon V \to D$ assigning a domain element $I(x)$ to each variable $x \in V$. The *satisfaction relation* $I \models \varphi$ is defined as follows:

- $I \models$ *true* and $I \not\models$ *false*;
- $I \models x \leq d$ if $I(x) \leq d$;
- $I \models x \geq d$ if $I(x) \geq d$;
- $I \models \varphi \wedge \psi$ if $I \models \varphi$ and $I \models \psi$;
- $I \models \varphi \vee \psi$ if $I \models \varphi$ or $I \models \psi$.

If $I \models \varphi$ holds, we say that I *satisfies* φ.

It can be easily seen that the literals $+0$ and $-(n-1)$ are superfluous since the inequalities $x \geq 0$ and $x \leq n-1$ are always satisfied. We call these literals *trivial*, contrary to the *non-trivial* literals $+1, \ldots, +(n-1)$ and $-0, \ldots, -(n-2)$. Without loss of generality, it is sufficient to only consider patterns and clausal languages with non-trivial literals.

A clausal pattern $P = (+a_1, \ldots, +a_p, -b_1, \ldots, -b_q)$ is said to be

- *Horn* if $p \leq 1$,
- *dual Horn* if $q \leq 1$,
- *bijunctive* if $p + q \leq 2$ and *binary* if $p + q = 2$,
- *d-valid* if $d \geq \min(P^+)$ or $d \leq \max(P^-)$,
- *positive* if $p > 0$ and $q = 0$ (i.e., $P = P^+$),
- *negative* if $p = 0$ and $q > 0$ (i.e., $P = P^-$),
- *monotone* if it is positive or negative,
- *mixed* if $p > 0$ and $q > 0$.

Note that if $\vec{x}$ is any vector of variables such that $|\vec{x}| = |P|$, then the pattern P is d-valid if and only if $P(\vec{x})$ is satisfied by the constant assignment $I(x) = d$ for all $x \in \vec{x}$. A clausal language L is Horn, dual Horn, bijunctive, or d-valid, respectively, if every pattern in L has the corresponding property.

A *relation* R of arity k over a finite domain D is a subset $R \subseteq D^k$. A vector $m \in R$ belonging to R is denoted by $m = (m[1], \dots, m[k])$, where $m[i]$ is the value of m at the i-th coordinate. Each L-formula φ gives rise to a corresponding relation

$$\mathrm{Sol}(\varphi(x_1, \dots, x_k)) \quad = \quad \{(I(x_1), \dots, I(x_k)) \mid I \models \varphi\}$$

produced by the set of assignments I satisfying φ.

Let $R \subseteq D^k$ be a relation of arity k over the domain D and $f\colon D \to D$ be a unary function. The *image* of the relation R under the function f is the set of vectors

$$f(R) \quad = \quad \{(f(m[1]), \dots, f(m[k])) \mid m \in R\}$$

of arity k. We say that the relation R is *closed* under f if the inclusion $f(R) \subseteq R$ holds.

In the sequel we need the image of clausal patterns under unary functions. Let P be a clausal pattern of length k over the domain D and $f\colon D \to D$ be a unary function. The *image* of the pattern P under the function f is the relation

$$f(P) \quad = \quad \{f(m) \mid m \in \mathrm{Sol}(P(x_1, \dots, x_k))\}.$$

We say that a pattern P is *closed* under f if the inclusion $f(P) \subseteq \mathrm{Sol}(P(x_1, \dots, x_k))$ holds. In this case, f is called an *endomorphism* (or unary polymorphism) of P. A unary function f is called an endomorphism of a clausal language L if the inclusion $f(P) \subseteq \mathrm{Sol}(P(x_1, \dots, x_k))$ holds for each $P \in L$.

Let $f\colon D \to D$ be a unary function defined on the domain D. We denote the range of f by $f(D)$ and the k-fold *composition* of f by f^k (defined recursively as $f^1 = f$ and $f^{k+1} = f^k \circ f$).

3 Constraint Satisfaction Problems

Given a clausal language L, the *constraint satisfaction problem* over L is defined as follows.

Problem: $\textsc{csp}(V, D, L)$
Input: An L-formula φ over a finite totally ordered domain D and variables V.
Question: Is φ satisfiable?

If the domain D and variables V are implicitly clear, we write $\textsc{csp}(L)$ instead of $\textsc{csp}(V, D, L)$.

In this section we will show that, in order to classify the complexity of problems $\textsc{csp}(L)$, it is sufficient to do this for languages which contain all unary patterns and are closed under taking subpatterns.

Lemma 1 *Let L be a clausal language, such that the positive literal $+a$ for some $a \in \{1, \dots, n-1\}$ (or the negative literal $-b$ for some $b \in \{0, \dots, n-2\}$) does not occur in any pattern of L. Then there exist a clausal language L' on the domain $D \smallsetminus \{a\}$ (or $D \smallsetminus \{b\}$), such that* $\textsc{csp}(L)$ *and* $\textsc{csp}(L')$ *are polynomial-time equivalent.*

Proof: Assume that the positive literal $+a$ for some $a \in \{1, \dots, n-1\}$ does not appear in any pattern in L (the case of $-b$ is very similar). Let $D' = D \setminus \{a\}$ be the new domain. For every pattern $P \in L$, let P' be a pattern on D' obtained from P by replacing each literal $-a$ (if there are any) by $-(a-1)$. Set $L' = \{P' \mid P \in L\}$. We claim that $\mathrm{CSP}(L)$ and $\mathrm{CSP}(L')$ are polynomial-time equivalent.

If φ is an instance of $\mathrm{CSP}(L)$ then let the instance φ' of $\mathrm{CSP}(L')$ be obtained by replacing each pattern P in it by the corresponding pattern P' while keeping the same variables. If I is a satisfying assignment to φ and $I(x) = a$ holds for some variable x then this assignment, but with $I(x) = a-1$, would still satisfy all clauses in φ (we use here the linearity of order and the fact that the literal $x \geq a$ is not present in φ). Doing this for every variable x with $I(x) = a$, we get a satisfying assignment to φ which does not use a. It is easy to see that this assignment will also be a solution to φ'. On the other hand, if I' is a satisfying assignment to φ' then it is also a solution to φ.

The reduction from $\mathrm{CSP}(L')$ to $\mathrm{CSP}(L)$ is similar. For every pattern P' in L', fix a pattern $P \in L$ such that P' is obtained from P by replacing each literal $-a$ (if there are any) by $-(a-1)$. The reduction is simply the inverse mapping to the previous one. □

Remark 2 It follows from the definitions that if f is an endomorphism of L and I is a satisfying assignment to an instance of $\mathrm{CSP}(L)$ then $f \circ I$ also satisfies the instance. In particular, every satisfiable instance of $\mathrm{CSP}(L)$ has a solution using only values from $f(D)$.

Lemma 3 *Let L be a clausal language that has an endomorphism f which is not a permutation. Then there exist a clausal language L' on a smaller domain such that* $\mathrm{CSP}(L)$ *and* $\mathrm{CSP}(L')$ *are polynomial-time equivalent.*

Proof: It is easy to see that for any k the iterated function f^k is also an endomorphism and that for some k the mapping f^k acts identically on its range, i.e., that $f^k(a) = a$ holds for every $a \in f^k(D)$. Without loss of generality, we may assume that already the unary function f has this property. Since f is not a permutation, we can find an element $b \in D$ which is not in the range of f, i.e., such that $b \notin f(D)$. Let $f(b) = a$ for some $a \in f(D)$. We have either $a < b$ or $a > b$. We perform the proof only for the case $a < b$, since the other one is similar.

Let $D = \{0, \dots, n-1\}$ be the domain. We need a case analysis corresponding to the position of b in D. If $b = n-1$ then every satisfiable instance of $\mathrm{CSP}(L)$ has a solution over $\{0, \dots, n-2\}$ according to Remark 2. So we can simply remove $n-1$ from D and all occurrences of $n-1$ from the patterns in L. Clearly this leads to an equivalent problem over a smaller domain.

Let $b < n-1$. For every pattern $P \in L$, let P'' be a pattern obtained from P by replacing each literal $+b$ (if there are any) by $+(b+1)$. Set $L'' = \{P'' \mid P \in L\}$. We claim that $\mathrm{CSP}(L)$ and $\mathrm{CSP}(L'')$ are polynomial-time equivalent. Since f is also an endomorphism of L'', following Lemma 1 every satisfiable instance of $\mathrm{CSP}(L'')$ has a satisfying assignment $I \colon V \to f(D)$ using only values from the range of f. Clearly, changing all literals of the form $x \geq b$ to $x \geq b+1$ does not affect the satisfiability of instances of $\mathrm{CSP}(L)$. □

By Lemmas 1 and 3, in order to classify the complexity of problems $\mathrm{CSP}(L)$, it is sufficient to consider only clausal languages L whose all endomorphisms are permutations and such that each non-trivial literal is present in some pattern in L. To further restrict the class of clausal languages, we need to temporarily extend the type of constraints under consideration by allowing not only constraints given by patterns from L, but also *constant* constraints, i.e., constraints of the form

$x = a$ where x is a variable and a is an element from D. For a clausal language L, let $\text{CSP}_c(L)$ denote the extended problem $\text{CSP}(L)$ in which arbitrary constant constraints are also allowed in instances.

With respect to Corollary 4.8 in [BJK05] (or by Theorem 3.7 in [BJK99]), $\text{CSP}(L)$ and $\text{CSP}_c(L)$ are polynomial-time equivalent, whenever all endomorphisms of a clausal language L are permutations.

Definition 4 We call a clausal language L **SU-closed**, if it is closed under taking subpatterns and it contains all non-trivial unary patterns, i.e., if it satisfies the following conditions:

1. If $P \in L$ and P' is a subpattern of P then $P' \in L$.
2. $(+d) \in L$ for all $d \in \{1, \ldots, n-1\}$.
3. $(-d) \in L$ for all $d \in \{0, \ldots, n-2\}$.

Proposition 5 *For any clausal language L over a fixed domain D there exists an SU-closed clausal language L', such that $\text{CSP}(L)$ and $\text{CSP}(L')$ are polynomial-time equivalent and L' can be constructed from L in time polynomial in the size of L.*

Proof: First, we describe a polynomial-time algorithm which finds a non-surjective endomorphism of L or shows that such an endomorphism does not exist. Let

$$P \quad = \quad (+a_1, \ldots, +a_p, -b_1, \ldots, -b_q)$$

be a pattern in L and $R_P = \text{Sol}(P(x_1, \ldots, x_{p+q}))$ be the corresponding relation on D. By definition, a unary function f on D is *not* an endomorphism of R_P if and only if there exists a vector $m = (m_1, \ldots, m_p, m'_1, \ldots, m'_q) \in R_P$, such that $f(m) \notin R_P$, i.e., if and only if the following conditions hold:

1. There exists an i, such that $i \in \{1, \ldots, p\}$ and $m_i \geq a_i$ or $i \in \{1, \ldots, q\}$ and $m'_i \leq b_i$.
2. For all $j \in \{1, \ldots, p\}$ we have $f(m_j) < a_j$.
3. For all $j \in \{1, \ldots, q\}$ we have $f(m'_j) > b_j$.

Clearly, for any given f, the above conditions can be verified in linear time. Since D is fixed, the number of possible unary functions on D is a constant. So the algorithm simply checks for each possible non-surjective unary function f, whether f is an endomorphism of relation R_P for each pattern $P \in L$.

Using the above algorithm, we can detect non-surjective endomorphisms of L. If f is such an endomorphism and b is an element absent from the range of f then we follow the (linear-time) procedures from the proofs of Lemmas 1 and 3 to obtain an equivalent language over domain $D \setminus \{b\}$. We perform these steps until all endomorphisms of the obtained language are permutations, i.e., at most $|D|$ times.

By Lemmas 1 and 3, we may assume that all endomorphisms of L are permutations and that each non-trivial literal is present in some pattern in L. We will show that the language L' obtained from L by adding all unary patterns together with all subpatterns of the patterns in L is the required language. Note that every unary pattern is a subpattern of some pattern in L.

We will show that $\text{CSP}(L)$ and $\text{CSP}_c(L')$ are polynomial-time equivalent. This, together with obvious inclusions $\text{CSP}(L) \subseteq \text{CSP}(L') \subseteq \text{CSP}_c(L')$, will prove the proposition.

As we noticed before this proposition, $\text{CSP}(L)$ is polynomial-time equivalent to $\text{CSP}_c(L)$. It is easy to show that $\text{CSP}_c(L)$ and $\text{CSP}_c(L')$ are polynomial-time equivalent. Assume that $P = (+a_1, +a_2, \ldots, +a_p, -b_1, \ldots, -b_q) \in L$ and $Q = (+a_2, \ldots, +a_p, -b_1, \ldots, -b_q)$. Clearly the clause $(x_2 \geq a_2 \vee \ldots \vee x_p \geq a_p \vee x_{p+1} \leq b_1 \vee \ldots \vee x_{p+q} \leq b_q)$ is obtained by resolution from the clauses $x_1 = 0$ and $(x_1 \geq a_1 \vee x_2 \geq a_2 \vee \ldots \vee x_p \geq a_p \vee x_{p+1} \leq b_1 \vee \ldots \vee x_{p+q} \leq b_q)$. Similarly, all patterns obtained from patterns of L by removing one literal can be added to L, and the new problem will be equivalent to $\text{CSP}_c(L)$. Continuing this procedure, we will eventually obtain that $\text{CSP}_c(L)$ and $\text{CSP}_c(L')$ are polynomial-time equivalent, which implies that $\text{CSP}(L)$ and $\text{CSP}_c(L')$ are equivalent. □

According to Proposition 5, we can get rid of redundant values in the domain. Thus, we can restrict the $\text{CSP}(L)$ problems to the case where the clausal language L is closed under taking subpatterns and it contains all unary patterns. **Therefore we consider only SU-closed clausal languages L in the sequel**.

4 Complexity of CSP Problems

Some polynomial cases have already been identified in the literature before without the use of clausal patterns and languages. The satisfiability problems for L-formulas built from Horn, dual Horn, and bijunctive clausal languages, respectively, were studied in the framework of many-valued logics [Häh01] as well as in the one of constraints [CJJK00, JC95].

Proposition 6 ([CJJK00, Häh01, JC95]) $\text{CSP}(L)$ *for a Horn, dual Horn, or bijunctive clausal language L is decidable in polynomial time.*

This result does not prove all polynomial-time decidable cases of $\text{CSP}(L)$, but it shows that three polynomial-time decidable classes of Boolean constraint satisfaction problems extend to finite totally ordered domains, provided that we use the clausal patterns paradigm.

As in the Boolean case, it turns out that it is sufficient to examine the patterns of length at most 3 that can be obtained from L (or "implemented" by L, see [CKS01]). For this reason we introduce the notion of 3-saturation of a clausal language which is based on the concept of *pattern resolution.* Let $P = (v_1, \ldots, v_p, +a)$ and $Q = (-b, v'_1, \ldots, v'_q)$ be two patterns, such that P contains a positive literal $+a$ and Q a negative literal $-b$, satisfying $b < a$. Then the pattern $R = (v_1, \ldots, v_p, v'_1, \ldots, v'_q)$ is called a *resolvent* of P and Q. We say that R is obtained from P and Q by resolution.

Definition 7 Let L be an SU-closed clausal language, i.e. a clausal language closed under taking subpatterns and containing all unary patterns. The **3-saturation** of L, denoted by $\widehat{L}$, contains all patterns that can be constructed inductively from L by the following rules:

1. If $P \in L$ and $|P| \leq 3$, then $P \in \widehat{L}$ (introduction).

2. If P and Q are patterns in $\widehat{L}$ such that $|P| + |Q| \leq 5$, then all resolvents of P and Q are in $\widehat{L}$ (restricted resolution).

Note that due to the restriction, only resolvents with a length at most 3 are considered in the second condition.

Remark 8 Observe that $\widehat{L}$ is an SU-closed language and that it can be computed from L in finite time.

Proposition 9 *$\textsc{csp}(\widehat{L})$ is reducible to $\textsc{csp}(L)$ in polynomial time.*

Proof: For each formula $\varphi(\vec{x})$ over $\widehat{L}$ there exists a formula $\varphi'(\vec{x},\vec{y})$ over L, such that φ' can be obtained from φ by means of resolution in polynomial time. It follows from resolution for many-valued regular logic (and is easy to see) that the formula φ is satisfiable if and only if φ' is. □

We need to analyze carefully which parts of the $\textsc{csp}(L)$ problem are polynomial-time decidable and which are NP-complete. To this end we need the concept of disjoint PN-pair.

Definition 10 We call two patterns P and N a **PN-pair** if P is positive ($P = P^+$), N is negative ($N = N^-$), one of them has length 2, and the other has length 3. A PN-pair (P, N) is called **disjoint** when $\max(N) < \min(P)$ holds, otherwise it is **overlapping**.

We will perform a reduction from the following well-known problem, which is also known as a constraint satisfaction problem over the set of Boolean relations $\{or_3^+, or_2^-\}$ with $or_3^+ = \{0,1\}^3 \smallsetminus \{000\}$ and $or_2^- = \{0,1\}^2 \smallsetminus \{11\}$. Its NP-completeness follows from Schaefer's result [Sch78]. The problem can also be described as $\textsc{csp}(L)$, where L is the clausal language $\{(+1,+1,+1), (-0,-0)\}$.

Problem: MONOTONE SAT
Input: A Boolean formula $\varphi = c_1 \wedge \cdots \wedge c_k$ in conjunctive normal form, where each clause c_i has either three positive literals or two negative literals.
Question: Is φ satisfiable?

Proposition 11 *If the 3-saturation $\widehat{L}$ contains a disjoint PN-pair of patterns then $\textsc{csp}(L)$ is* NP-*complete.*

Proof: Let $P = (+a_1, +a_2, +a_3)$ be a positive pattern of length 3 and $N = (-b_1, -b_2)$ be the negative pattern of length 2 in $\widehat{L}$ that constitute the disjoint PN-pair. Let $d = \max(N)$. For each Boolean L-formula φ we construct an $\widehat{L}$-formula φ' in the following way. For each clause $or_3^+(x,y,z)$ in φ we put the clause $P(x,y,z)$ in φ' and for each clause $or_2^-(x,y)$ in φ we put the clause $N(x,y)$ in φ'. Since P and N are disjoint it is easy to check that φ is satisfiable if and only if φ' is (intuitively values smaller than or equal to d will be identified to 0, while the others will be identified to 1).

The proof for a positive pattern P of length 2 and a negative pattern N of length 3 follows from the previous case by the duality principle. □

5 Dichotomy Result

We must analyze now the cases when the 3-saturation $\widehat{L}$ does not contain a disjoint PN-pair. This will be done by a case analysis.

Case 1: All positive patterns in $\widehat{L}$ have length 1, or dually all negative patterns in $\widehat{L}$ have length 1.

This case corresponds to the Horn and dual Horn cases.

Proposition 12 *Let L be an SU-closed clausal language. If all positive (negative) patterns in $\widehat{L}$ have length 1 then both L and $\widehat{L}$ are Horn (dual Horn).*

Proof: Suppose that L is not Horn, i.e., there exists a pattern M with at least two positive literals. The clausal language L is closed under subpatterns, therefore there must be a positive pattern $P = (+p_1, +p_2)$ in L. Since $|P| \leq 3$ holds, we have that $P \in \widehat{L}$, what is a contradiction with the assumption. The result for dual Horn follows from the duality principle. □

Tractability of the cases considered in Proposition 12 follows from Proposition 6.

Case 2: The 3-saturation $\widehat{L}$ contains at least one positive and one negative pattern of length greater or equal to 2.

Case 2 splits to the following sub-cases.

Case 2.1: All patterns in $\widehat{L}$ have length smaller or equal to 2.

If all patterns in $\widehat{L}$ have length smaller or equal to 2 then L is bijunctive and $\text{CSP}(L)$ is tractable by Proposition 6.

Case 2.2: There exists at least one pattern $M \in \widehat{L}$ of length $|M| = 3$.

We need to analyze the case when there exists a pattern $M \in \widehat{L}$ of length $|M| = 3$. This is the most involved part of the proof. To this end we need to introduce the notions of $[a, b]$-valid, $[a, b]$-satisfiable, and $[a, b]$-unsatisfiable patterns on an interval $[a, b]$.

Definition 13 A pattern M of length k is called **$[a, b]$-valid** if *every* assignment $I : V \to [a, b]$ satisfies the clause $M(x_1, \ldots, x_k)$, i.e., if every assignment whose range is restricted to the interval $[a, b]$ satisfies $M(x_1, \ldots, x_k)$. M is called **$[a, b]$-satisfiable** if the clause $M(x_1, \ldots, x_k)$ is satisfiable by an assignment $I : V \to [a, b]$, i.e., by an assignment whose range is restricted to the interval $[a, b]$. M is called **$[a, b]$-unsatisfiable** when there is *no* satisfying assignment $I : V \to [a, b]$ of the clause $M(x_1, \ldots, x_k)$, i.e., when there is no assignment restricted to the interval $[a, b]$ that satisfies $M(x_1, \ldots, x_k)$.

It is clear that on a given interval $[a, b]$ a pattern M is $[a, b]$-valid, or $[a, b]$-satisfiable, or $[a, b]$-unsatisfiable. Moreover, $[a, b]$-validity implies $[a, b]$-satisfiability. The notion of $[a, b]$-satisfiability of M means that every satisfying assignment of $M(x_1, \ldots, x_k)$ can be restricted to the values from the interval $[a, b]$, contrary to the $[a, b]$-unsatisfiable patterns. It should also be clear that a pattern M can be $[a, b]$-unsatisfiable even if the clause $M(x_1, \ldots, x_k)$ is satisfiable by an assignment I whose range is not included in the interval $[a, b]$.

Definition 14 Let L be a clausal language, such that $\widehat{L}$ contains a positive binary and a negative binary pattern. We call the values

$$p_{\max} = \max\{\min(P) \mid P \in \text{pos}_2 \widehat{L}\} \qquad \text{and} \qquad q_{\min} = \min\{\max(N) \mid N \in \text{neg}_2 \widehat{L}\}$$

Propagate L:	$(C \vee x \leq v) \wedge (x \leq u) \rightarrow (x \leq u)$	if $u \leq v$
Propagate R:	$(C \vee x \geq u) \wedge (x \geq v) \rightarrow (x \geq v)$	if $u \leq v$
Restrict L:	$(C \vee x \leq u) \wedge (x \geq v) \rightarrow C \wedge (x \geq v)$	if $u < v$
Restrict R:	$(C \vee x \geq v) \wedge (x \leq u) \rightarrow C \wedge (x \leq u)$	if $u < v$
Contradiction:	$(x \leq u) \wedge (x \geq v) \rightarrow \bot$	if $u < v$

Figure 1: Formula simplification rules

the **markers** of the binary positive and negative patterns in the saturation $\widehat{L}$, respectively. We also call $P_* = (+p_{\max}, +p_{\max})$ and $N_* = (-q_{\min}, -q_{\min})$ the **extremal patterns**.

Observe that in Case 2.2 these markers are well-defined, since there are binary positive and negative patterns in the 3-saturation $\widehat{L}$.

Our goal is to present a polynomial-time algorithm that decides the satisfiability of an L-formula φ. This algorithm starts with the following pre-processing step. From φ we construct a new L-formula φ' by exhaustive application of the five rules presented in Figure 1. Since these rules are confluent and terminating, we can apply them in an arbitrary order and always obtain the unique normal form φ'.

It is clear that φ' can be computed from φ in polynomial time. The formulas φ and φ' are logically equivalent, therefore they also have the same satisfiability property. Observe that every variable in φ' occurs at most once in a unary positive (resp. negative) clause. Moreover, if a variable x occurs both in a positive clause $(x \geq u)$ and in a negative clause $(x \leq v)$, then these two clauses are compatible, i.e., $u \leq v$ holds, and both clauses are satisfied by two assignments $I(x) = u$ and $I'(x) = v$. These observations will be used to justify the correctness of the polynomial-time algorithms developed in the sequel. Moreover, note that the patterns of all constraints occurring in φ' belong to L, since L is closed under subpatterns.

Case 2.2 splits once more. We need to perform a case analysis on the position of markers of the binary monotone patterns in $\widehat{L}$.

Case 2.2.1: $p_{\max} \leq q_{\min}$. Observe that when the relation $p_{\max} \leq q_{\min}$ holds then there cannot be a disjoint PN-pair of patterns in L.

Proposition 15 *Let L be an SU-closed clausal language, such that $\widehat{L}$ contains a positive binary, a negative binary, and a ternary pattern. If the markers satisfy $p_{\max} \leq q_{\min}$, then* CSP(L) *is decidable in polynomial time.*

Proof: Let φ be an L-formula. Transform φ to a new L-formula φ' by an exhaustive application of the simplification rules from Figure 1. Observe that all non-unit clauses from φ' are d-valid for every $d \in [p_{\max}, q_{\min}]$. Indeed, for sake of contradiction, suppose that there exists a clause c in φ, which is not d-valid for some $d \in [p_{\max}, q_{\min}]$. Let M_c be the pattern associated to c. Since it is not d-valid, all literals of M_c must be of the form $+p$ with $p > d \geq p_{\max}$ or $-q$ with $q < d \leq q_{\min}$. M_c cannot be positive since otherwise we can produce from M_c by taking subpatterns a positive binary pattern $P = (+p_1, +p_2) \in \widehat{L}$ with $p_i > p_{\max}$ for each i, but this contradicts the definition of $p_{\max}$ and the fact that the SU-closed clausal language L is closed under subpatterns. Dually, M_c cannot be negative since otherwise we could produce a negative binary pattern $N = (-q_1, -q_2) \in \widehat{L}$ with $q_i < p_{\max} \leq q_{\min}$, which contradicts the definition of $q_{\min}$ and the SU-closedness of L. If M_c is mixed, we can derive by taking subpatterns a pattern $M' = (+p, -q) \in \widehat{L}$ with $q < p_{\max} < p$. There exists by definition a pattern $P_1 = (+p_{\max}, +p_1) \in \widehat{L}$ for some $p_1 \geq p_{\max}$. We obtain by two resolution steps from M' and P_1 the pattern $Q = (+p, +p) \in \widehat{L}$, which contradicts the definition of $p_{\max}$, since $p > p_{\max}$ holds.

The algorithm to find a satisfying assignment for φ proceeds as follows. First, it transforms φ to φ' by an exhaustive application of the rules from Figure 1. If φ' contains the empty clause $\perp$ then φ is unsatisfiable. Otherwise, assign first all variables to an arbitrary but fixed $d \in [p_{\max}, q_{\min}]$. This initial assignment $I(x) = d$ for all $x \in V$ satisfies all non-unit clauses, according to the previous observation, but not necessarily the unit ones. If a variable x appears in a unit clause $(x \geq a)$ (or $x \leq b$) not satisfied by $I(x) = d$, then change the value of x to a (resp. b).

We need to prove that these assignment modifications do not alter the satisfaction of the non-unit clauses. Suppose that x occurs in φ' in a unit positive clause $(x \geq v)$ which is not not satisfied by $I(x) = d$, i.e., we have $v > d$. There are three types of clauses in φ' where the variable x can occur, namely a negative unit clause $(x \leq u)$, as well as the non-unit clauses $(C \vee x \geq u)$ with $u > v$ and $(C \vee x \leq u)$ with $u \geq v$. A negative unit clause $(x \leq u)$ must be compatible with $(x \geq v)$, since φ' does not contain the empty clause, thus it is satisfied by $I(x) = v$. The assignment $I(x) = d$ does not contribute to the satisfaction of the clause $(C \vee x \geq u)$ since $d < v \leq u$ holds. If $I(x) = d$ satisfies the clause $(C \vee x \leq u)$ then $I(x) = v$ satisfies it, too. Therefore changing the assignment from $I(x) = d$ to $I(x) = v$ will not affect the satisfiability of the non-unit clauses. □

Case 2.2.2: $q_{\min} < p_{\max}$. In this case the CSP(L) problem can be substantially simplified and studied on the restricted interval $[q_{\min}, p_{\max}]$.

Lemma 16 *Let L be an SU-closed clausal language such that $\widehat{L}$ contains a positive binary, a negative binary, and a ternary pattern. If $\widehat{L}$ does not contain a disjoint PN-pair and the markers satisfy the condition $q_{\min} < p_{\max}$, then every pattern $M \in L$ of length $|M| \geq 3$ is $[q_{\min}, p_{\max}]$-valid.*

Proof: Observe first that both extremal patterns $N_* = (-q_{\min}, -q_{\min})$ and $P_* = (+p_{\max}, +p_{\max})$ belong to $\widehat{L}$. Indeed, by definition of $q_{\min}$ and $p_{\max}$, there exist patterns $N = (-q_1, -q_{\min}) \in \widehat{L}$ and $P = (+p_1, +p_{\max}) \in \widehat{L}$, where $q_1 \leq q_{\min} < p_{\max} \leq p_1$ holds. We can produce both extremal patterns P_* and N_* by two resolution steps from P and N.

Suppose that there exists a pattern $M \in L$ of length $|M| \geq 3$, which is not $[q_{\min}, p_{\max}]$-valid. This means that M contains only the literals $-b$ with $b < p_{\max}$ and $+a$ with $a > q_{\min}$.

Let M be positive, i.e., $M = (+a_1, \ldots, +a_k)$, where $k \geq 3$ and $a_i > q_{\min}$ holds for all i. Then the subpattern $M_3^+ = (+a_1, +a_2, +a_3)$ belongs to $\widehat{L}$, and together with $N_* = (-q_{\min}, -q_{\min})$ forms a disjoint PN-pair, which is a contradiction with the assumption. Let M be negative, i.e., $M = (-b_1, \ldots, -b_k)$, where $k \geq 3$ and $b_i < p_{\max}$ holds for all i. Then, the subpattern $M_3^- = (-b_1, -b_2, -b_3)$ belongs to $\widehat{L}$, and together with $P_* = (+p_{\max}, +p_{\max})$ forms a disjoint PN-pair, which is once more a contradiction with the assumption. Let $M = (+a_1, \ldots, +a_k, -b_1, \ldots, -b_\ell)$ be mixed, where $k + \ell \geq 3$, $a_i > q_{\min}$, and $b_j < p_{\max}$ hold for all i, j. Then either the subpattern $M_3 = (+a_1, +a_2, -b_1)$ or the subpattern $M_3 = (+a_1, -b_1, -b_2)$ belongs to $\widehat{L}$. Thus, by repeated resolution of M_3 and P_* we can produce the positive pattern $M_3' = (+a_1, +p_{\max}, +p_{\max})$, where $a_1 > q_{\min}$ and $p_{\max} > q_{\min}$ hold. This leads to a contradiction as before. □

Lemma 17 *Let L be an SU-closed clausal language such that $\widehat{L}$ contains a positive binary, a negative binary, and a ternary pattern. If $\widehat{L}$ does not contain a disjoint PN-pair and the markers satisfy the condition $q_{\min} < p_{\max}$, then every binary pattern in L is $[q_{\min}, p_{\max}]$-satisfiable.*

Proof: Recall that both extremal patterns $N_* = (-q_{\min}, -q_{\min})$ and $P_* = (+p_{\max}, +p_{\max})$ belong to $\widehat{L}$.

Let $P = (+a, +b) \in L$ be a positive pattern with $a \leq b$. P belongs to $\widehat{L}$. If $a \leq q_{\min}$ holds then P is $[q_{\min}, p_{\max}]$-valid and therefore also $[q_{\min}, p_{\max}]$-satisfiable. The pattern $Q_* = (-q_{\min}, +p_{\max})$ is a resolvent of the extremal patterns $P_* = (+p_{\max}, +p_{\max})$ and $N_* = (-q_{\min}, -q_{\min})$ since $q_{\min} < p_{\max}$ holds. If $q_{\min} < a \leq p_{\max}$ holds then we obtain $Q' = (+b, +p_{\max})$ by resolution from P and Q_*. By resolution from Q' and N_* we get $Q'' = (+b, -q_{\min})$. Another resolution step between Q' and Q'' gives $Q_\dagger = (+b, +b)$. If $b > p_{\max}$ then this contradicts the definition of $p_{\max}$, therefore we must have $b \leq p_{\max}$ and $[a, b] \subseteq [q_{\min}, p_{\max}]$. In this case, P is $[q_{\min}, p_{\max}]$-satisfiable. The same result holds by duality for negative binary patterns in L.

Let $M = (-a, +b) \in L$. If $a \geq p_{\max}$ or $b \leq q_{\min}$ hold then M is $[q_{\min}, p_{\max}]$-valid and therefore also $[q_{\min}, p_{\max}]$-satisfiable. If $a < p_{\max}$ and $b > q_{\min}$ hold then we need a supplementary case analysis. If $a \geq q_{\min}$ and $b \leq p_{\max}$ then the interval $[a, b]$ or $[b, a]$, depending whether $a < b$ or $b \leq a$, is included in $[q_{\min}, p_{\max}]$ and therefore M is $[q_{\min}, p_{\max}]$-satisfiable. If $a < q_{\min}$ then we can produce by resolution from M and $N_* = (-q_{\min}, -q_{\min})$ the pattern $Q = (-a, -q_{\min})$, followed by two additional resolution steps as above, which give $Q_\dagger = (-a, -a)$, what contradicts the definition of $q_{\min}$. If $b > p_{\max}$ holds then we can produce by resolution from M and $P_* = (+p_{\max}, +p_{\max})$ the pattern $Q' = (+p_{\max}, +b)$, followed by two additional resolution steps as above, which give $Q_\dagger = (+b, +b)$, what contradicts the definition of $p_{\max}$. □

Proposition 18 *Let L be an SU-closed clausal language such that $\widehat{L}$ contains a positive binary, a negative binary, and a ternary pattern. If $\widehat{L}$ does not contain a disjoint PN-pair and the markers satisfy the condition $q_{\min} < p_{\max}$, then* CSP(L) *is decidable in polynomial time.*

Proof: The algorithm to find a satisfying assignment for φ in polynomial time proceeds as follows. First, it transforms φ to φ' by an exhaustive application of the rules from Figure 1. If φ' contains the empty clause $\perp$ then φ is unsatisfiable. Otherwise, construct the sub-formula φ'_2 consisting of all $[q_{\min}, p_{\max}]$-satisfiable clauses from φ' which are not $[q_{\min}, p_{\max}]$-valid. According to Lemmas 16 and 17, φ'_2 is bijunctive, i.e., it contains only unit and binary clauses, and therefore its satisfiability is decidable in polynomial time. Find a satisfying assignment for φ'_2 on the interval $[q_{\min}, p_{\max}]$. The remaining clauses in φ' are either "big" clauses, i.e., of length 3 or more, binary clauses, or unit clauses. The first two types are $[q_{\min}, p_{\max}]$-valid, so the computed satisfying assignment satisfies them. What remains are $[q_{\min}, p_{\max}]$-unsatisfiable unit clauses.

As in the proof of Proposition 15, we change the values of the concerned variables, setting $I(x) = v$ if either $x \leq v$ or $x \geq v$ is a $[q_{\min}, p_{\max}]$-unsatisfiable clause in φ'. Once more, by construction of φ' for the same reason as in the proof of Proposition 15, this change does not alter the satisfaction of the already satisfied clauses. Suppose that x occurs in φ' in a unit positive clause $(x \geq v)$ which is not satisfied by $I(x) = d$, i.e., we have $v > d$. There are three types of clauses in φ' where the variable x can occur, namely a negative unit clause $(x \leq u)$, as well as the non-unit clauses $(C \vee x \geq u)$ with $u > v$ and $(C \vee x \leq u)$ with $u \geq v$. A negative unit clause $(x \leq u)$ must be compatible with $(x \geq v)$, since φ' does not contain the empty clause, thus it is satisfied by $I(x) = v$. The assignment $I(x) = d$ does not contribute to the satisfaction of the clause $(C \vee x \geq u)$

since $d < v \leq u$ holds. If $I(x) = d$ satisfies the clause $(C \vee x \leq u)$ then $I(x) = v$ satisfies it, too. Therefore changing the assignment from $I(x) = d$ to $I(x) = v$ will not affect the satisfiability of the non-unit clauses. □

To the best of our knowledge, the tractable cases identified by means of Proposition 18 are new. The following theorem follows from Propositions 11, 12, 15, and 18.

Theorem 19 (Dichotomy Theorem) *Let L be an SU-closed clausal language. If the saturation $\widehat{L}$ does not contain a disjoint PN-pair then* CSP(L) *is decidable in polynomial time, otherwise* CSP(L) *is NP-complete.*

Example 20 Consider the SU-closed clausal language

$$\begin{aligned} L \;=\; \{&(-0),\ (-1),\ (+1),\ (+2),\\ &(+1,-0),\ (+1,+1),\ (-1,-1),\ (+2,+2),\\ &(+1,+1,+1)\} \end{aligned}$$

over the 3-element domain $D = \{0, 1, 2\}$. The only endomorphism of L is the identity. The 3-saturation $\widehat{L}$ of L is then equal to $L \cup \{(+2,-1),\ (+1,+2),\ (+1,-1)\}$. There is no disjoint PN-pair in $\widehat{L}$, therefore CSP(L) is polynomial-time decidable.

6 Concluding Remarks

We presented a complexity analysis of the constraint satisfaction problems over finite totally ordered domains, based on the new concept of clausal patterns. We derived decidable conditions for CSP problems implying either NP-completeness or polynomial-time satisfiability. In fact, our polynomial-time cases generalize the previously known characterization through Horn, dual Horn, bijunctive, 0- and 1-valid Boolean relations. Not surprisingly the 3-saturation concept allows us to decide algorithmically the dividing condition between the tractable and intractable instances. The result is a dichotomy theorem applicable to the CSP problems over totally ordered domains of arbitrary cardinality.

By combining tractability results from the paper with the algebraic approach, it is possible to obtain many more tractable (non-clausal) CSPs. We leave this as an open problem — to characterize tractable (non-clausal) constraint languages that can be generated from clausal ones. This would actually amount to describing tractable clausal languages algebraically. Our CSP problems are somewhat orthogonal to the relationally defined ones. On one hand we obtained a dichotomy theorem, on the other hand relations can express much more constraints than clauses, so our classification is coarser. Another possible extension of our research is to consider clausal patterns over a partially rather than totally ordered domain. We believe that the results presented in this paper can be generalized to this more general case.

Acknowledgment: We sincerely thank Julien Demouth for proofreading the paper and pointing out several writing errors.

References

[BJK99] A. Bulatov, P. Jeavons, and A. Krokhin. Constraint satisfaction problems and finite algebras. Technical Report TR-4-99, Oxford University Computing Laboratory, 1999.

[BJK05] A. Bulatov, P. Jeavons, and A. Krokhin. Classifying the complexity of constraints using finite algebras. *SIAM Journal on Computing*, 2005. To appear.

[Bul02] A. A. Bulatov. A dichotomy theorem for constraints on a three-element set. In *Proceedings 43rd Symposium on Foundations of Computer Science (FOCS 2002), Vancouver (British Columbia, Canada)*, pages 649–658, November 2002.

[Bul03] A. A. Bulatov. Tractable conservative constraint satisfaction problems. In *Proceedings 18th IEEE Symposium on Logic in Computer Science (LICS 2003), Ottawa (Canada)*, pages 321–330, June 2003.

[CJJK00] D. Cohen, P. Jeavons, P. Jonsson, and M. Koubarakis. Building tractable disjunctive constraints. *Journal of the Association for Computing Machinery*, 47(5):826–853, 2000.

[CKS01] N. Creignou, S. Khanna, and M. Sudan. *Complexity Classifications of Boolean Constraint Satisfaction Problems*, volume 7 of *SIAM Monographs on Discrete Mathematics and Applications*. SIAM, Philadelphia (PA), 2001.

[Dal02] V. Dalmau. Constraint satisfaction problems in non-deterministic logarithmic space. In P. Widmayer, F. Triguero Ruiz, R. Morales Bueno, M. Hennessy, S. Eidenbenz, and R. Conejo, editors, *Proceedings 29th International Conference on Automata, Languages, and Programming (ICALP 2002), Malaga (Spain)*, volume 2380 of *Lecture Notes in Computer Science*, pages 414–425. Springer-Verlag, July 2002.

[FV98] 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.

[Häh01] R. Hähnle. Complexity of many-valued logics. In *Proceedings 31st IEEE International Symposium on Multiple-Valued Logic (ISMVL 2001), Warsaw (Poland)*, pages 137–148. IEEE Computer Society, May 2001.

[HN04] P. Hell and J. Nešetřil. *Graphs and Homomorphisms*. Oxford University Press, 2004.

[JC95] P. Jeavons and M. C. Cooper. Tractable constraints on ordered domains. *Artificial Intelligence*, 79(2):327–339, 1995.

[KV00] P. G. Kolaitis and M. Y. Vardi. Conjunctive-query containment and constraint satisfaction. *Journal of Computer and System Science*, 61(2):302–332, 2000.

[Lad75] R. E. Ladner. On the structure of polynomial time reducibility. *Journal of the Association for Computing Machinery*, 22(1):155–171, 1975.

[Sch78] T. J. Schaefer. The complexity of satisfiability problems. In *Proceedings 10th Symposium on Theory of Computing (STOC'78), San Diego (California, USA)*, pages 216–226, 1978.