The Complexity of Satisfiability Problems: Refining Schaefer's Theorem*

Eric Allender[1], Michael Bauland[2], Neil Immerman[3], Henning Schnoor[4], and Heribert Vollmer[5]

[1] Department of Computer Science, Rutgers University, Piscataway, NJ 08855, allender@cs.rutgers.edu

[2] Theoretische Informatik, Universität Hannover, Appelstr. 4, 30167 Hannover, Germany. bauland@thi.uni-hannover.de

[3] Department of Computer and Information Science, University of Massachusetts, Amherst, MA 01003, immerman@cs.umass.edu

[4] Institut für Informatik, Christian-Albrechts-Universität zu Kiel, 24098 Kiel, Germany schnoor@ti.informatik.uni-kiel.de

[5] **Corresponding author**: Theoretische Informatik, Universität Hannover, Appelstr. 4, 30167 Hannover, Germany. vollmer@thi.uni-hannover.de. Phone: +49 511 762 19768. Fax: +49 511 762 19606.

* Supported in part by DFG grants Vo 630/5-1/2, Vo 630/6-1, and NSF Grants CCF-0514155, DMS-0652582, CCF-0830133, CCF-0832787, and CCF-0514621.

Abstract. Schaefer proved in 1978 that the Boolean constraint satisfaction problem for a given constraint language is either in P or is NP-complete, and identified all tractable cases. Schaefer's dichotomy theorem actually shows that there are at most two constraint satisfaction problems, up to polynomial-time isomorphism (and these isomorphism types are distinct if and only if P $\neq$ NP). We show that if one considers AC^0 isomorphisms, then there are exactly six isomorphism types (assuming that the complexity classes $NP, P, \oplus L, NL$, and L are all distinct). A similar classification holds for quantified constraint satisfaction problems.

Keywords: Computational Complexity, Constraint Satisfaction Problem, Propositional Satisfiability, Clone Theory

1 Introduction

In 1978, Schaefer classified the Boolean constraint satisfaction problem and showed that, depending on the allowed relations in a propositional formula, the problem is either in P or is NP-complete [Sch78]. This famous "dichotomy theorem" does not consider the fact that different problems in P have quite different complexity, and there is now a well-developed complexity theory to classify different problems in P. Furthermore, in Schaefer's original work (and in the many subsequent simplified presentations of his theorem [CKS01]) it is already apparent that certain classes of constraint satisfaction problems are either trivial (the 0-valid and 1-valid relations) or are solvable in NL (the bijunctive relations) or $\oplus$L (the affine relations), whereas for other problems (the Horn and anti-Horn relations) he provides only a reduction to problems that are complete for P. Is this a *complete* list of complexity classes that can arise in the study of constraint satisfaction problems? Given the amount of attention that the dichotomy theorem has received, it is surprising that no paper had addressed the question of how to refine Schaefer's classification beyond some steps in this direction in Schaefer's original paper (see [Sch78, Theorem 5.1]), prior to a preliminary version of the current work [ABI$^+$05]. Subsequently, there have been some efforts to refine the classification of *non-Boolean* constraint satisfaction problems [Dal05,LT07].

Our own interest in this question grew out of the observation that there is at least one other fundamental complexity class that arises naturally in the study of Boolean constraint satisfaction problems that does *not* appear in the list (NL, $\oplus$L, P) of nontrivial feasible cases identified by Schaefer. This is the class SL (symmetric logspace) that has recently been shown by Reingold to coincide with deterministic logspace [Rei05]. (Theorem 5.1 of [Sch78] does already present examples of constraint satisfaction problems that are complete for SL.) Are there other classes that arise in this way? We give a negative answer to this question. If we examine constraint satisfaction problems using AC^0 reducibility $\leq_m^{\mathrm{AC}^0}$, then we are able to show that the following list of complexity classes is exhaustive: Every constraint satisfaction problem not solvable in coNLOGTIME is isomorphic to the standard complete set for one of the classes NP, P, $\oplus$L, NL, or L under isomorphisms computable and invertible in AC^0. (Definitions of notions such as coNLOGTIME and AC^0 can be found in Section 2.)

Our proofs rely heavily on universal algebra (in particular, the theory of *polymorphisms* and *clones*) and its consequences concerning the complexity of constraints. An introduction to this connection can be found in [Pip97], and in the surveys [BCRV03,BCRV04]. A more general introduction to clones on non-Boolean domains is [Lau06]. A thorough introduction, including applications to the area of constraint satisfaction problems (including CSPs with infinite domains) can be found in [CJ06]. In the next section we recall some of the relevant definitions and state, as facts, some of the required results in this area. One of the contributions of this paper is to point out that, in order to obtain a complete classification of constraint satisfaction problems (up to AC^0 isomorphism) it is necessary to go beyond the partition of constraint satisfaction problems given by their polymorphisms, and examine the constraints themselves in more detail.

2 Preliminaries

We assume that the reader is familiar with the standard complexity classes NP, P, NL, L, and AC^0; detailed definitions and background material on these classes can be found in [Vol99,HO02]. The class $\oplus$L is perhaps less familiar; it contains decision problems that can be solved by nondeterministic logspace machines, where an instance is accepted if the number of accepting paths of the machine is even. The very small complexity class AC^0 consists of all languages that are accepted by alternating Turing machines running in logarithmic time and making $O(1)$ alternations; this is the class that is called ATIME-ALT$(\log n, 1)$ in the text by Vollmer [Vol99]. AC^0 also can be defined as the class of languages accepted by uniform polynomial-size constant-depth families of unbounded fan-in AND and OR gates. Alternatively, in the framework of finite model theory, AC^0 is equivalent in expressive power to first-order logic, and thus AC^0 is sometimes denoted by FO. In particular, AC^0 reductions are also known as FO-translations. See the text by Immerman [Imm99] for more on the connection between first-order logic and AC^0. The smallest complexity class to which we will refer is coNLOGTIME, a small subclass of AC^0. coNLOGTIME consists of the complements of all languages accepted by nondeterministic Turing machines (having random access to their input tape) that run for time $O(\log n)$ on inputs of length n.

An n-ary Boolean relation is a subset of $\{0,1\}^n$. For a set V of variables, a constraint application C is an application of an n-ary Boolean relation R to an n-tuple of variables $(x_1, \ldots, x_n)$ from V. An assignment $I\colon V \to \{0,1\}$ *satisfies* the constraint application $R(x_1, \ldots, x_n)$ if and only if $(I(x_1), \ldots, I(x_n)) \in R$. We may use a propositional formula, $\psi(x_1, \ldots, x_n)$, to define the relation $R_\psi = \{(\alpha_1, \ldots, \alpha_n) \mid \psi(\alpha_1, \ldots, \alpha_n) = 1\}$.

A *constraint language* is a finite set of nonempty Boolean relations. The Boolean *Constraint Satisfaction Problem* over a constraint language Γ (CSP(Γ)) is the question of whether a given set φ of Boolean constraint applications using relations from Γ is simultaneously satisfiable, i.e., if there exists an assignment $I\colon V \to \{0,1\}$, such that I satisfies every $C \in \varphi$. It is easy to see that the Boolean CSP over some language Γ is the same as satisfiability of conjunctive Γ-formulas. For example, consider 3SAT: a well-known restriction of the general satisfiability problem. 3SAT can be seen to be the CSP over the language $\Gamma_{3\mathrm{SAT}} = \{(x_1 \vee x_2 \vee x_3), (\overline{x_1} \vee x_2 \vee x_3), (\overline{x_1} \vee \overline{x_2} \vee x_3), (\overline{x_1} \vee \overline{x_2} \vee \overline{x_3})\}$.

We now summarize some of the results concerning the very useful connection between the complexity of the CSP and universal algebra, referring the reader to [Pip97,BCRV03,BCRV04] for details.

A class of Boolean functions is a *clone*, if it contains all projection functions and is closed under composition. $[B]$ denotes the smallest clone containing the set of Boolean functions B. Equivalently, $[B]$ is the set of Boolean functions that can be calculated by Boolean circuits using only gates for functions from B.

The set of clones forms a lattice with $[B] \sqcap [C] = [B] \cap [C]$ and $[B] \sqcup [C] = [B \cup C]$. Emil Post [Pos20,Pos41] identified all clones and their inclusion structure (Figure 1). A description of the clones and a list of bases for each one can

be found in Table 1. For a description of the properties of the clones arising here, see, e.g., [BCRV03].

Recall that we are interested in studying the complexity of CSP(Γ) for various sets of Boolean relations, Γ. The following definition connects such a set of Boolean relations, Γ, to the clone, Pol(Γ).

Definition 2.1. *A k-ary relation R is **closed** or **invariant** under an n-ary Boolean function f, and f is a **polymorphism** of R, if and only if for all $x_1, \ldots, x_n \in R$ with $x_i = (x_i[1], x_i[2], \ldots, x_i[k])$, we have*

$$(f(x_1[1], \ldots, x_n[1]), f(x_1[2], \ldots, x_n[2]), \ldots, f(x_1[k], \ldots, x_n[k])) \in R.$$

We denote the set of all polymorphisms of R by Pol(R)*, and for a set Γ of Boolean relations we define* $\mathrm{Pol}(\Gamma) = \{f \mid f \in \mathrm{Pol}(R) \text{ for every } R \in \Gamma\}$*. For a set B of Boolean functions,* $\mathrm{Inv}(B) = \{R \mid B \subseteq \mathrm{Pol}(R)\}$ *is the set of **invariants** of B.*

Recall that a *conjunctive query* over Γ is a relation of the form

$$R(x_1, \ldots, x_n) \quad \equiv \quad \exists y_1, \ldots, y_m R_1(z_{1,1}, \ldots, z_{1,n_1}) \wedge \cdots \wedge R_k(z_{k,1}, \ldots, z_{k,n_k})$$

where $R_i \in \Gamma$ and $z_{i,j} \in \{x_1, \ldots, x_n, y_1, \ldots, y_m\}$. Define COQ($\Gamma$) to be the set of all conjunctive queries over Γ. Define the **co-clone** generated by Γ: $\langle\Gamma\rangle = \mathrm{COQ}(\Gamma \cup \{=\})$. In other words, $\langle\Gamma\rangle$ is the set of relations that can be expressed as a primitive positive formula with clauses from Γ.

For any set of relations, Γ, every projection is a polymorphism of Γ, and the composition of two polymorphisms is a polymorphism. Thus Pol(Γ) is a clone. It is similarly not hard to see that Inv(B) is always a co-clone. The following fact summarizes the properties of the Galois connection between the lattices of clones and co-clones, see, e.g., [Gei68]. It first has been applied in the context of constraint satisfaction problems by Jeavons, Cohen, and Gyssens in [JCG97].

Fact 2.2 *For any sets of Boolean functions, B, B', and Boolean relations, S, S', the following hold:*

1. $\mathrm{Inv}(\mathrm{Pol}(S)) = \langle S\rangle$
2. $\mathrm{Pol}(\mathrm{Inv}(B)) = \langle B\rangle$
3. $S \subseteq S' \quad \Rightarrow \quad \mathrm{Pol}(S') \subseteq \mathrm{Pol}(S)$
4. $B \subseteq B' \quad \Rightarrow \quad \mathrm{Inv}(B') \subseteq \mathrm{Inv}(B)$

The concept of relations closed under certain Boolean functions is interesting, because many properties of Boolean relations can be equivalently formulated using this terminology. For example, a set of relations can be expressed by Horn-formulas if and only if every relation in the set is closed under the binary AND function. Horn is one of the properties that ensures the corresponding satisfiability problem to be tractable. More generally, tractability of formulas over a given set of relations only depends on the set of its polymorphisms.

Name	Definition	Base
BF	All Boolean functions	$\{\vee, \wedge, \neg\}$
R_0	$\{f \in BF \mid f$ is 0-reproducing $\}$	$\{\wedge, \oplus\}$
R_1	$\{f \in BF \mid f$ is 1-reproducing $\}$	$\{\vee, \leftrightarrow\}$
R_2	$R_1 \cap R_0$	$\{\vee, x \wedge (y \leftrightarrow z)\}$
M	$\{f \in BF \mid f$ is monotonic $\}$	$\{\vee, \wedge, 0, 1\}$
M_1	$M \cap R_1$	$\{\vee, \wedge, 1\}$
M_0	$M \cap R_0$	$\{\vee, \wedge, 0\}$
M_2	$M \cap R_2$	$\{\vee, \wedge\}$
S_0^n	$\{f \in BF \mid f$ is 0-separating of degree $n\}$	$\{\rightarrow, \text{dual}(h_n)\}$
S_0	$\{f \in BF \mid f$ is 0-separating $\}$	$\{\rightarrow\}$
S_1^n	$\{f \in BF \mid f$ is 1-separating of degree $n\}$	$\{x \wedge \overline{y}, h_n\}$
S_1	$\{f \in BF \mid f$ is 1-separating $\}$	$\{x \wedge \overline{y}\}$
S_{02}^n	$S_0^n \cap R_2$	$\{x \vee (y \wedge \overline{z}), \text{dual}(h_n)\}$
S_{02}	$S_0 \cap R_2$	$\{x \vee (y \wedge \overline{z})\}$
S_{01}^n	$S_0^n \cap M$	$\{\text{dual}(h_n), 1\}$
S_{01}	$S_0 \cap M$	$\{x \vee (y \wedge z), 1\}$
S_{00}^n	$S_0^n \cap R_2 \cap M$	$\{x \vee (y \wedge z), \text{dual}(h_n)\}$
S_{00}	$S_0 \cap R_2 \cap M$	$\{x \vee (y \wedge z)\}$
S_{12}^n	$S_1^n \cap R_2$	$\{x \wedge (y \vee \overline{z}), h_n\}$
S_{12}	$S_1 \cap R_2$	$\{x \wedge (y \vee \overline{z})\}$
S_{11}^n	$S_1^n \cap M$	$\{h_n, 0\}$
S_{11}	$S_1 \cap M$	$\{x \wedge (y \vee z), 0\}$
S_{10}^n	$S_1^n \cap R_2 \cap M$	$\{x \wedge (y \vee z), h_n\}$
S_{10}	$S_1 \cap R_2 \cap M$	$\{x \wedge (y \vee z)\}$
D	$\{f \mid f$ is self-dual$\}$	$\{x\overline{y} \vee x\overline{z} \vee \overline{y}\overline{z}\}$
D_1	$D \cap R_2$	$\{xy \vee x\overline{z} \vee y\overline{z}\}$
D_2	$D \cap M$	$\{xy \vee yz \vee xz\}$
L	$\{f \mid f$ is linear$\}$	$\{\oplus, 1\}$
L_0	$L \cap R_0$	$\{\oplus\}$
L_1	$L \cap R_1$	$\{\leftrightarrow\}$
L_2	$L \cap R$	$\{x \oplus y \oplus z\}$
L_3	$L \cap D$	$\{x \oplus y \oplus z \oplus 1\}$
V	$\{f \mid f$ is constant or a n-ary OR function$\}$	$\{\vee, 0, 1\}$
V_0	$[\{\vee\}] \cup [\{0\}]$	$\{\vee, 0\}$
V_1	$[\{\vee\}] \cup [\{1\}]$	$\{\vee, 1\}$
V_2	$[\{\vee\}]$	$\{\vee\}$
E	$\{f \mid f$ is constant or a n-ary AND function$\}$	$\{\wedge, 0, 1\}$
E_0	$[\{\wedge\}] \cup [\{0\}]$	$\{\wedge, 0\}$
E_1	$[\{\wedge\}] \cup [\{1\}]$	$\{\wedge, 1\}$
E_2	$[\{\wedge\}]$	$\{\wedge\}$
N	$[\{\neg\}] \cup [\{0\}] \cup [\{1\}]$	$\{\neg, 1\}$
N_2	$[\{\neg\}]$	$\{\neg\}$
I	$[\{\text{id}\}] \cup [\{0\}] \cup [\{1\}]$	$\{\text{id}, 0, 1\}$
I_0	$[\{\text{id}\}] \cup [\{0\}]$	$\{\text{id}, 0\}$
I_1	$[\{\text{id}\}] \cup [\{1\}]$	$\{\text{id}, 1\}$
I_2	$[\{\text{id}\}]$	$\{\text{id}\}$

Table 1: List of all closed classes of Boolean functions, and their bases
$h_n(x_1, \ldots, x_{n+1}) = \bigvee_{i=1}^{n+1} x_1 \wedge x_2 \wedge \cdots \wedge x_{i-1} \wedge x_{i+1} \wedge \cdots \wedge x_{n+1}$

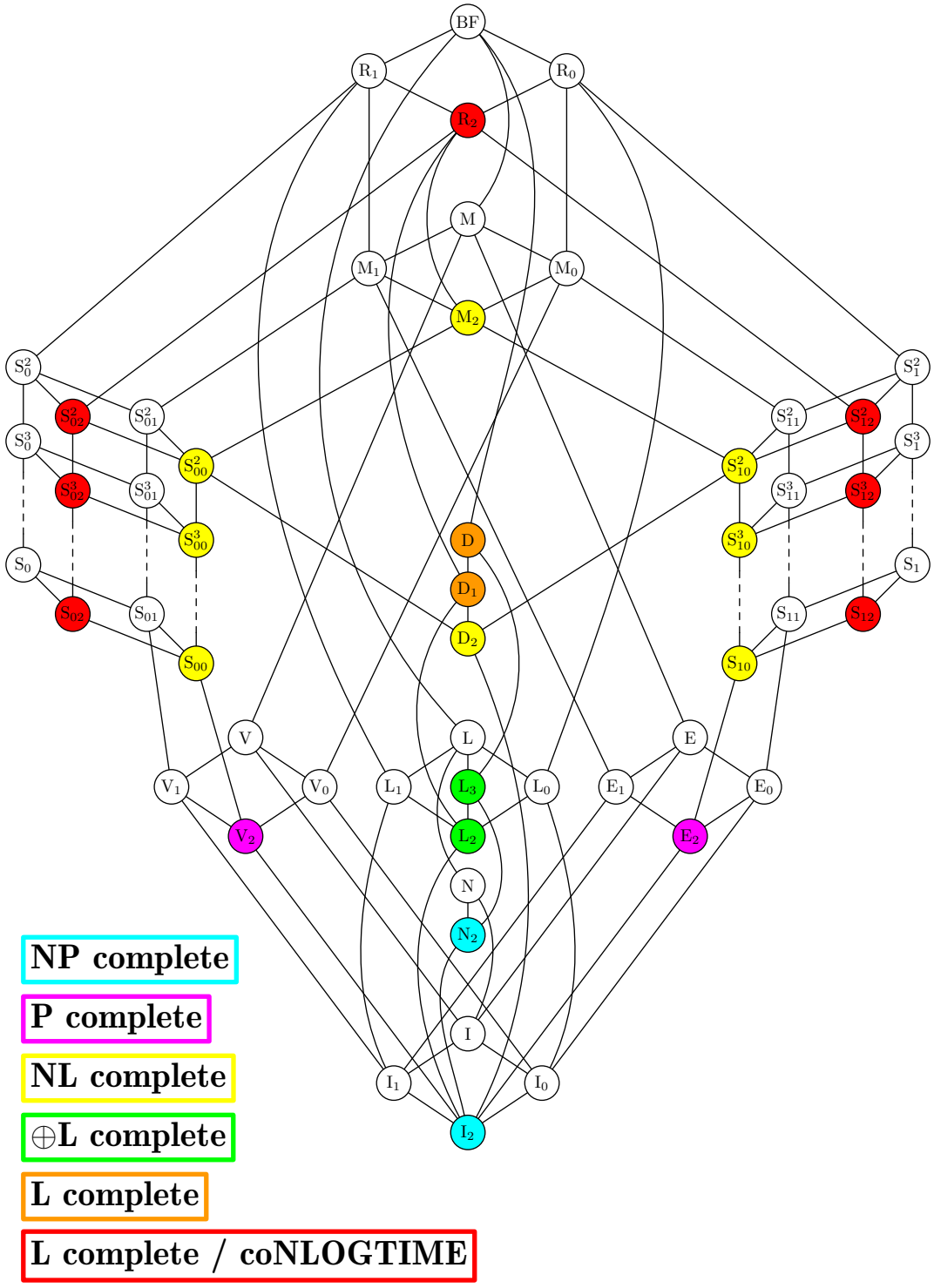

Fig. 1. Graph of all closed classes of Boolean functions

Corollary 2.3. *Let Γ_1 and Γ_2 be sets of Boolean relations such that Γ_1 is finite and* $\mathrm{Pol}(\Gamma_2) \subseteq \mathrm{Pol}(\Gamma_1)$. *Then* $\mathrm{CSP}(\Gamma_1) \leq_m^{\mathrm{p}} \mathrm{CSP}(\Gamma_2)$.

Proof. Since $\mathrm{Pol}(\Gamma_2) \subseteq \mathrm{Pol}(\Gamma_1)$, we know from Fact 2.2 (parts 1 and 4) and from the definition of co-clone, that $\Gamma_1 \subseteq \mathrm{COQ}(\Gamma_2 \cup \{=\})$. Thus, in polynomial time, we can translate any element of $\mathrm{CSP}(\Gamma_1)$ to an equivalent element of $\mathrm{CSP}(\Gamma_2)$. (The equality constraints can be removed in polynomial time by choosing one representative variable for those variables constrained to be equal to it.)

The most general constraint language, Γ, is such that $\mathrm{Pol}(\Gamma)$ is the minimal clone, i.e., the clone containing only projection functions. In this case, $\langle\Gamma\rangle$ is the set of all Boolean relations. For any such Γ, $\mathrm{CSP}(\Gamma)$ is NP-complete. For example, it can be shown that $\mathrm{Pol}(\Gamma_{3\mathrm{SAT}})$ contains only the projections, and hence 3SAT is NP-complete.

As we have seen in the above corollary, the complexity of the CSP for a given constraint language is determined by the set of its polymorphisms. At least this is the case when considering gross classifications of complexity (such as whether a problem is in P or is NP-complete). However, when we examine finer complexity classifications, such as determining the circuit complexity of a constraint satisfaction problem, then the set of polymorphisms of a constraint language Γ does *not* completely determine the complexity of $\mathrm{CSP}(\Gamma)$, as can easily be seen in the following important example:

Example 2.4. Let $\Gamma_1 = \{\overline{x}, x\}$, $\Gamma_2 = \Gamma_1 \cup \{=\}$. It is obvious that $\mathrm{Pol}(\Gamma_1) = \mathrm{Pol}(\Gamma_2)$; the set of polymorphisms is the clone R_2. Formulas over Γ_1 only contain clauses of the form x or $\overline{x}$ for some variable x, whereas in Γ_2, we additionally have the binary equality predicate. We will now see that $\mathrm{CSP}(\Gamma_1)$ has very different complexity than $\mathrm{CSP}(\Gamma_2)$.

Satisfiability of a Γ_1-formula φ can be decided in coNLOGTIME. (Such a formula is unsatisfiable if and only if for some variable x, both x and $\overline{x}$ are clauses.)

In contrast, $\mathrm{CSP}(\Gamma_2)$ is complete for L under $\leq_m^{\mathrm{AC}^0}$ reductions: The complement of the graph accessibility problem (GAP) for undirected graphs, which is known to be complete for L [Rei05], can be reduced to $\mathrm{CSP}(\Gamma_2)$. Let $G = (V, E)$ be a finite, undirected graph, and s, t vertices in V. For every edge $(v_1, v_2) \in E$, add a constraint $v_1 = v_2$. Also add $\overline{s}$ and t. It is obvious that there exists a path in G from s to t if and only if the resulting formula is not satisfiable. In fact, it is easy to see that $\mathrm{CSP}(\Gamma_2)$ is not only hard for L, but it also lies within L so it is complete for L under $\leq_m^{\mathrm{AC}^0}$ reductions.

The lesson to learn from this example is that the usual reduction among constraint satisfaction problems arising from the same co-clone is not an $\leq_m^{\mathrm{AC}^0}$ reduction. The following lemma summarizes the main relationships.

Lemma 2.5. *Let Γ_1 and Γ_2 be sets of relations over a finite set, where Γ_1 is finite and* $\mathrm{Pol}(\Gamma_2) \subseteq \mathrm{Pol}(\Gamma_1)$. *Then* $\mathrm{CSP}(\Gamma_1) \leq_m^{\mathrm{AC}^0} \mathrm{CSP}(\Gamma_2 \cup \{=\}) \leq_m^{\log} \mathrm{CSP}(\Gamma_2)$.

Proof. Since the local replacement from Corollary 2.3 can be computed in AC^0, this establishes the first reducibility relation (note that variables are implicitly existentially quantified and therefore the quantifiers do not need to be written).

For the second reduction, we need to eliminate all of the =-constraints. We do this by identifying variables x_{i_1} and x_{i_2} if there is an =-path from x_{i_1} to x_{i_2} in the formula. By [Rei05], this can be computed in logspace. □

3 Classification

The following is our main result on the complexity of the Boolean constraint satisfaction problem.

Theorem 3.1. *Let Γ be a finite set of Boolean relations.*

- *If* $\mathrm{I}_0 \subseteq \mathrm{Pol}(\Gamma)$ *or* $\mathrm{I}_1 \subseteq \mathrm{Pol}(\Gamma)$*, then every constraint formula over Γ is satisfiable, and therefore* $\mathrm{CSP}(\Gamma)$ *is trivial.*
- *If* $\mathrm{Pol}(\Gamma) \in \{\mathrm{I}_2, \mathrm{N}_2\}$*, then* $\mathrm{CSP}(\Gamma)$ *is* $\leq_m^{\mathrm{AC}^0}$*-complete for* NP*.*
- *If* $\mathrm{Pol}(\Gamma) \in \{\mathrm{V}_2, \mathrm{E}_2\}$*, then* $\mathrm{CSP}(\Gamma)$ *is* $\leq_m^{\mathrm{AC}^0}$*-complete for* P*.*
- *If* $\mathrm{Pol}(\Gamma) \in \{\mathrm{L}_2, \mathrm{L}_3\}$*, then* $\mathrm{CSP}(\Gamma)$ *is* $\leq_m^{\mathrm{AC}^0}$*-complete for* $\oplus$L*.*
- *If* $\mathrm{S}_{00} \subseteq \mathrm{Pol}(\Gamma) \subseteq \mathrm{S}_{00}^2$ *or* $\mathrm{S}_{10} \subseteq \mathrm{Pol}(\Gamma) \subseteq \mathrm{S}_{10}^2$ *or* $\mathrm{Pol}(\Gamma) \in \{\mathrm{D}_2, \mathrm{M}_2\}$*, then* $\mathrm{CSP}(\Gamma)$ *is* $\leq_m^{\mathrm{AC}^0}$*-complete for* NL*.*
- *If* $\mathrm{Pol}(\Gamma) \in \{\mathrm{D}_1, \mathrm{D}\}$*, then* $\mathrm{CSP}(\Gamma)$ *is* $\leq_m^{\mathrm{AC}^0}$*-complete for* L*.*
- *If* $\mathrm{S}_{02} \subseteq \mathrm{Pol}(\Gamma) \subseteq \mathrm{R}_2$ *or* $\mathrm{S}_{12} \subseteq \mathrm{Pol}(\Gamma) \subseteq \mathrm{R}_2$*, then either* $\mathrm{CSP}(\Gamma)$ *is in* coNLOGTIME*, or* $\mathrm{CSP}(\Gamma)$ *is complete for* L *under* $\leq_m^{\mathrm{AC}^0}$*. There is an algorithm deciding which case occurs.*

Theorem 3.1 is a refinement of Theorem 5.1 from [Sch78] and Theorem 6.5 from [CKS01]. It is immediate from a look at Figure 1 that this covers all cases. The proof follows from the lemmas in the following subsections. First, we mention a corollary:

Corollary 3.2. *For any set of relations Γ,* $\mathrm{CSP}(\Gamma)$ *is* AC^0*-isomorphic either to $0\Sigma^*$ or to the standard complete set for one of the following complexity classes:* $\mathrm{NP}, \mathrm{P}, \oplus\mathrm{L}, \mathrm{NL}, \mathrm{L}$.

Proof. It is immediate from Theorem 3.1 that if $\mathrm{CSP}(\Gamma)$ is not in AC^0, then it is complete for one of $\mathrm{NP}, \mathrm{P}, \mathrm{NL}, \mathrm{L}$, or $\oplus\mathrm{L}$ under $\leq_m^{\mathrm{AC}^0}$ reductions. By [Agr01] each of these problems is AC^0-isomorphic to the standard complete set for its class. On the other hand, if $\mathrm{CSP}(\Gamma)$ is solvable in AC^0, then it is an easy matter to reduce any problem $A \in \mathrm{AC}^0$ to $\mathrm{CSP}(\Gamma)$ via a length-squaring, invertible AC^0 reduction (by first checking if $x \in A$, and then using standard padding techniques to map x to a long satisfiable instance if $x \in A$, and mapping x to a long syntactically incorrect input if $x \notin A$). AC^0 isomorphism to $0\Sigma^*$ now follows, since any two sets that are reducible to each other via length-squaring, invertible AC^0 reductions are AC^0-isomorphic [ABI97]. □

3.1 Upper Bounds: Algorithms

First, we state results that are well known; see, e.g., [Sch78,BCRV04]:

Fact 3.3 *Let Γ be a Boolean constraint language.*

1. *If* $\mathrm{Pol}(\Gamma) \in \{\mathrm{I}_2, \mathrm{N}_2\}$, *then* $\mathrm{CSP}(\Gamma)$ *is* NP*-complete. Otherwise,* $\mathrm{CSP}(\Gamma) \in \mathrm{P}$.
2. $\mathrm{L}_2 \subseteq \mathrm{Pol}(\Gamma)$ *implies every relation in* Γ *is affine, thus* $\mathrm{CSP}(\Gamma) \in \oplus\mathrm{L}$.
3. $\mathrm{D}_2 \subseteq \mathrm{Pol}(\Gamma)$ *implies every relation in* Γ *is bijunctive, thus* $\mathrm{CSP}(\Gamma) \in \mathrm{NL}$.
4. $\mathrm{I}_0 \subseteq \mathrm{Pol}(\Gamma)$ *or* $\mathrm{I}_1 \subseteq \mathrm{Pol}(\Gamma)$ *implies every instance of* $\mathrm{CSP}(\Gamma)$ *is satisfiable by the all-0 or the all-1 tuple, and therefore* $\mathrm{CSP}(\Gamma)$ *is trivial.*

Lemma 3.4. *Let Γ be a constraint language.*

1. *If* $\mathrm{S}_{02} \subseteq \mathrm{Pol}(\Gamma)$ *or* $\mathrm{S}_{12} \subseteq \mathrm{Pol}(\Gamma)$, *then* $\mathrm{CSP}(\Gamma) \in \mathrm{L}$.
2. *If* $\mathrm{S}_{00} \subseteq \mathrm{Pol}(\Gamma)$ *or* $\mathrm{S}_{10} \subseteq \mathrm{Pol}(\Gamma)$, *then* $\mathrm{CSP}(\Gamma) \in \mathrm{NL}$.

Proof. First we consider the cases S_{00} and S_{02}. The following algorithm is based on the proof for Theorem 6.5 in [CKS01]. Observe that there is no finite set Γ such that $\mathrm{Pol}(\Gamma) = \mathrm{S}_{00}$ ($\mathrm{Pol}(\Gamma) = \mathrm{S}_{02}$, resp.). Therefore, $\mathrm{Pol}(\Gamma) \supseteq \mathrm{S}_{00}^k$ ($\mathrm{Pol}(\Gamma) \supseteq \mathrm{S}_{02}^k$, resp.) for some $k \geq 2$. Note that $\mathrm{Pol}(\{\mathrm{OR}^k, x, \overline{x}, \rightarrow, =\}) = \mathrm{S}_{00}^k$ (OR^k refers to the k-ary OR relation) and $\mathrm{Pol}(\{\mathrm{OR}^k, x, \overline{x}, =\}) = \mathrm{S}_{02}^k$ ([BRSV05]), and therefore by Lemma 2.5 we can assume w.l.o.g. $\Gamma = \{\mathrm{OR}^k, x, \overline{x}, \rightarrow, =\}$ ($\Gamma = \{\mathrm{OR}^k, x, \overline{x}, =\}$, resp.).

Now the algorithm works as follows: For a given formula φ over the relations mentioned above, consider every positive clause $x_{i_1} \vee \cdots \vee x_{i_k}$. The clause is satisfiable if and only if there is one variable in $\{x_{i_1}, \ldots, x_{i_k}\}$ which can be set to 1 without violating any of the $\overline{x}$ and $x \rightarrow y$ clauses (without violating any of the $\overline{x}$, resp.). For a variable $y \in \{x_{i_1}, \ldots, x_{i_k}\}$, this can be checked as follows:

For each clause $\overline{x}$, check if there is an $\rightarrow$-$=$-path ($=$-path, resp.) from y to x, by which we mean a sequence $yR_1z_1, z_1R_2z_2, \ldots, z_{m-1}R_mx$ for $R_i \in \{\rightarrow, =\}$ ($R_i \in \{=\}$, resp.). (This is just an instance of the GAP problem on *directed* graphs (*undirected* graphs, resp.), which is the standard complete problem for NL (L, resp.).) If one of these is the case, then y cannot be set to 1. Otherwise, we can set y to 1, and the clause is satisfiable. If a clause is shown to be unsatisfiable, reject. If no clause is shown to be unsatisfiable in this way, accept.

The S_{10}- and S_{12}-case are analogous; in these cases we have NAND instead of OR. □

Our final upper bound in this section is combined with a hardness result, and thus serves as a bridge to the next two sections. Note that the problems occurring here are essentially variants of determining whether a graph is 2-colorable.

Lemma 3.5. *Let Γ be a constraint language. If* $\mathrm{Pol}(\Gamma) \in \{\mathrm{D}_1, \mathrm{D}\}$, *then* $\mathrm{CSP}(\Gamma)$ *is* $\leq_m^{\mathrm{AC}^0}$*-complete for* L.

Proof. Note that $\mathrm{Pol}(\{\oplus\}) = \mathrm{D}$ and $\mathrm{Pol}(\{R\}) = \mathrm{D}_1$, where $R = x_1 \wedge (x_2 \oplus x_3)$. Thus by Lemmas 2.5 and 3.7, and Proposition 3.6, we can restrict ourselves to the cases where Γ consists of these relations only. The satisfiability problem for formulas that are conjunctions of clauses of the form x or $y \oplus z$ is complete for L by Problem 4.1 in Section 7 of [AG00], which proves completeness for the case $\mathrm{Pol}(\Gamma) = \mathrm{D}_1$ and thus proves membership in L for the case $\mathrm{Pol}(\Gamma) = \mathrm{D}$. It suffices to prove hardness in the case $\mathrm{Pol}(\Gamma) = \mathrm{D}$, by providing a reduction from $\mathrm{CSP}(\{x_1 \wedge (x_2 \oplus x_3)\})$.

This can easily be shown: For every clause x, introduce $x \oplus f$ for a new variable f, so now every clause is of the form $x \oplus y$. If the original formula is satisfiable, then the new one holds with the same assignment plus $f = 0$. If the new formula φ' is satisfiable, then there is some I such that $I \models \varphi'$. We know that $\overline{I} \models \varphi'$ as well, because $\oplus$ is closed under negation. Therefore, without loss of generality, $I(f) = 0$. Then $I \setminus \{f = 0\} \models \varphi$. Thus, the problem for formulas allowing x-clauses can be reduced to one not allowing them. Therefore, both cases are L-complete.

3.2 Removing the Equality Relation

Lemma 2.5 reveals that polymorphisms completely determine the complexity of a given constraint satisfaction problem only if the equality relation is contained in the corresponding constraint language. In Example 2.4 we saw that this question does lead to different complexity results. We now show that for most constraint languages, we can get equality "for free" and therefore the question of whether we have equality directly or not does not make a difference.

We say a constraint language Γ *can express* the relation $R(x_1, ..., x_n)$ if there is a formula $R_1(z_1^1, \ldots, z_{n_1}^1) \wedge \cdots \wedge R_l(z_1^l, \ldots, z_{n_l}^l)$, where $R_i \in \Gamma$ and $z_j^i \in \{y_1, \ldots, y_n, w_1, \ldots, w_r\}$ (the z_j^i's need not be distinct) such that for each assignment of values $(c_1, \ldots, c_n)$ to the variables $y_1, \ldots, y_n$, $R(c_1, ..., c_n)$ evaluates to TRUE if and only if there is an assignment of values to the variables $w_1, \ldots, w_r$ such that all R_i-clauses, with y_i replaced by c_i, evaluate to TRUE.

The following proposition is immediate.

Proposition 3.6. *Let Γ be a constraint language. If Γ can express the equality relation, then* $\mathrm{CSP}(\Gamma \cup \{=\}) \leq_m^{\mathrm{AC}^0} \mathrm{CSP}(\Gamma)$.

Lemma 3.7. *Let Γ be a finite set of Boolean relations where* $\mathrm{Pol}(\Gamma) \subseteq \mathrm{M}_2$, $\mathrm{Pol}(\Gamma) \subseteq \mathrm{L}$, *or* $\mathrm{Pol}(\Gamma) \subseteq \mathrm{D}$. *Then Γ can express the equality relation.*

Proof. The relation "$x \to y$" is invariant under M_2. Thus given any such Γ, by Corollary 2.3 we can construct "$x \to y$" with the help of new existentially quantified variables that do not appear anywhere else in the formula. Equality clauses between the variables x and y do not appear, since $x = y$ does not hold for every element of the relation (equality involving existentially quantified variables does not appear in the construction given in Corollary 2.3). Hence Γ can express $x = y$ with $x \to y \wedge y \to x$.

For the L-case, apply an analogous argument for the relation R^4_{even}, which consists of all 4-tuples with an even number of 1's. Note that $x = y$ is expressed by $R^4_{\text{even}}(z, z, x, y)$. If $\text{Pol}(\Gamma) \subseteq \text{D}$, then we can express $x \oplus y$, and thus we express equality by $x = y \iff (x \oplus z) \wedge (z \oplus y)$. □

As noted in Example 2.4, for some classes, the question of whether equality is contained in the constraint language or not does lead to different complexities, namely complete for L or contained in coNLOGTIME. We now show that there are no intermediate complexity classes arising in these cases. As we saw in the lemmas above, this only concerns constraint languages Γ such that $\text{Pol}(\Gamma) \supseteq \text{S}^m_{02}$ or $\text{Pol}(\Gamma) \supseteq \text{S}^m_{12}$ holds for some $m \geq 2$.

Lemma 3.8. *Let R be a relation such that $\text{Pol}(R) \supseteq \text{S}^m_{02}$ ($\text{Pol}(R) \supseteq \text{S}^m_{12}$, resp.). Let $S = \text{OR}^m$ ($S = \text{NAND}^m$, resp.). Then either $\text{CSP}(\{x, \overline{x}, S, R\}) \in$ coNLOGTIME or R can express equality (in which case $\text{CSP}(\{x, \overline{x}, S, R\})$ is hard for L under AC^0 reductions). There is an algorithm deciding which of the cases occurs.*

Proof. Since $\text{Pol}(\{x, \overline{x}, \text{OR}^m\}) = \text{S}^m_{02}$ ([BRSV05]), we know from Fact 2.2 that $R(x_1, \ldots, x_n)$ can be expressed using equality, positive and negative literals, and the m-ary OR predicate. Let φ be a representation of R in this form. We simplify φ as follows (without loss of generality, assume that R is not the empty relation):

0. Repeat steps $1 - 3$ as long as changes occur:
1. For any clause $x_1 = x_2$ where there is a clause consisting only of a single literal x_1, x_2, $\overline{x_1}$, or $\overline{x_2}$, remove the clause $x_1 = x_2$ and insert the corresponding literal for the other variable as well. Repeat until no such clause remains.
2. Remove variables from OR-clauses which appear as negative literals.
3. For an OR-clause containing variables connected with a path of equality clauses, remove all of them except one.

Note that this does not change the relation represented by the formula. After steps $1 - 3$ are executed, the simplified formula might now contain some literals that did not appear before, since an OR-clause can be reduced to a literal in step 2. Thus these steps need to be repeated. Each time the process is repeated, the number of literals increases or the arity of OR statements decreases. Thus the procedure will terminate after a finite number of repetitions. If no $=$-clause remains, then R can be expressed using only OR and literals and therefore leads to a CSP solvable in coNLOGTIME (a CSP-formula using only these relations is unsatisfiable if and only if there appear two contradictory variables or an OR-clause containing only variables which also appear as a negative literal).

Otherwise, let $x_1 = x_2$ be a remaining clause. We existentially quantify all variables in R except x_1 and x_2, and call the resulting relation R'. We claim that R' is the equality relation. Let $(x_1, x_2) \in R'$. Since $x_1 = x_2$ appears in the defining formula, $x_1 = x_2$ holds. For the other direction, let $x_1 = x_2$. We assign the value 0 to every existentially quantified variable that appears as a negative

literal, the same value as x_1 to every variable connected to x_1 via an $=$-path, and the value 1 to all others. Obviously, all literals are satisfied this way: Remember x_1 and x_2 do not appear as literals due to step 1, and there are no contradictory literals since R is nonempty. All equality clauses are satisfied because none of the variables appearing here also appear as literals. Let $(x_1 \vee \cdots \vee x_j)$ be a clause. None of these variables appear as negative literals due to step 2, and at most one of them can be $=$-connected to x_1 and x_2 due to step 3. Therefore, the assignment constructed above assigns 1 to at least one of the occurring variables, thus satisfying the formula. Hardness for L now follows with the same construction as in Example 2.4.

It is decidable which of these cases occurs: Since the only way to obtain equality is by existentially quantifying all variables except two, there is a finite number of combinations which can be easily verified by an algorithm. An analogous argument can be applied to the dual case $\mathrm{Pol}(R) \supseteq S_{12}^m$. □

Note that as in the proof of Lemma 3.4, if $\mathrm{Pol}(\Gamma) \supseteq \mathrm{S}_{02}$, then it follows that $\mathrm{Pol}(\Gamma) \supseteq \mathrm{S}_{02}^m$ for some $m \geq 2$. Hence the above result yields the following corollary:

Corollary 3.9. *Let Γ be a constraint language such that $\mathrm{S}_{02} \subseteq \mathrm{Pol}(\Gamma) \subseteq \mathrm{R}_2$ or $\mathrm{S}_{12} \subseteq \mathrm{Pol}(\Gamma) \subseteq \mathrm{R}_2$. Then either $\mathrm{CSP}(\Gamma) \in$ coNLOGTIME, or $\mathrm{CSP}(\Gamma)$ is complete for L under AC^0-reductions. There is an algorithm deciding which of these cases occurs.*

In the period since these results were first announced [ABI+05], a deeper understanding of this corollary has emerged, that is relevant for non-Boolean domains. Those Γ in Corollary 3.9 for which $\mathrm{CSP}(\Gamma) \in$ coNLOGTIME have what is known as the "finite duality" property [At05,Ros05]. It has been shown by Larose and Tesson [LT07] that CSPs that do not have finite duality are hard for L, while it is immediate that any CSP with finite duality is in coNLOGTIME. In addition, there is an algorithm to determine if Γ has the finite duality property larose.loten.tardif.

3.3 Lower Bounds: Hardness Results

One technique of proving hardness for constraint satisfaction problems is to reduce certain problems related to Boolean circuits to CSPs. In [Rei01], many decision problems regarding circuits were discussed. In particular, the "Satisfiability Problem for B Circuits" ($\mathrm{SAT}^{\mathrm{C}}(B)$) is very useful for our purposes here. $\mathrm{SAT}^{\mathrm{C}}(B)$ is the problem of determining if a given Boolean circuit with gates from B has an input vector on which it computes output "1".

Lemma 3.10. *Let Γ be a constraint language such that $\mathrm{Pol}(\Gamma) \in \{\mathrm{E}_2, \mathrm{V}_2\}$. Then $\mathrm{CSP}(\Gamma)$ is $\leq_m^{\mathrm{AC}^0}$-hard for* P.

Proof. Assume without loss of generality that Γ contains $=$. The proof of the general case then follows from Lemmas 2.5 and 3.7, and Proposition 3.6.

A relation can be expressed by a Horn (dual Horn, resp.) formula if and only if it is invariant under E_2 (V_2, resp.). It is well known that the satisfiability problems for Horn and anti-Horn formulas are P-complete under $\leq_m^{\log}$ reductions. We give a proof for the anti-Horn case showing hardness under $\leq_m^{\mathrm{AC}^0}$ reductions. (Membership in P follows directly from Schaefer's work.) The proof uses the standard idea of simulating each gate in a Boolean circuit with Boolean constraints expressing the function of each gate. We show $\mathrm{SAT}^C(\mathrm{S}_{11}) \leq_m^{\mathrm{AC}^0} \mathrm{CSP}(\Gamma)$. The result then follows from [Rei01] plus the observation that his hardness result holds under $\leq_m^{\mathrm{AC}^0}$. Let C be a $\{(x \wedge (y \vee z), c_0\}$-circuit. For each gate $g \in C$, introduce a new variable x_g. Now, introduce constraint clauses as follows:

1. Let g be a c_0-gate. Then add a constraint $\overline{x_g}$ (i.e., $x_g = 0$).
2. Let g be an $x \vee (y \wedge z)$-gate, and let g_x, g_y, g_z be the predecessor gates of g. Then introduce a constraint $x_g \rightarrow (x_{g_x} \wedge (x_{g_y} \vee x_{g_z}))$ (this can be expressed as a conjunction of two anti-Horn clauses as follows: $(\overline{x_g} \vee x_{g_x}) \wedge (\overline{x_g} \vee x_{g_y} \vee x_{g_z})$).
3. For the output-gate g, add a constraint x_g.

By construction, the resulting constraint φ is an anti-Horn-formula. Thus all relations are closed under V_2.

We claim $C \in \mathrm{SAT}^{\mathrm{C}}$ if and only if $\varphi \in \mathrm{CSP}(\Gamma)$.

Let $C \in \mathrm{SAT}^{\mathrm{C}}$. Now, assign to all variables in the constraint the value the corresponding gate in the circuit has when given the satisfying assignment to the input gates. That is, we are assuming that $C(\alpha_1, \ldots, \alpha_n) = 1$. Assign to any x_g in φ the value $\mathrm{val}_g(\alpha_1, \ldots, \alpha_n)$ (which is the value of the gate g when $(\alpha_1, \ldots, \alpha_n)$ is given as input for C). Obviously, all introduced constraint clauses are satisfied with this variable assignment.

Let $\varphi \in \mathrm{CSP}(\Gamma)$. Assign to all input gates of the circuit the corresponding value of the satisfying assignment for φ. It can easily be shown that for all $g \in C$, $\mathrm{val}(g) \geq x_g$ holds. Since this is true for the output gate as well, and the clause x_g (for $g \in C$ the output-gate of the circuit) exists in φ, the circuit value is 1. For the Horn case, a dual argument can be applied.

Lemma 3.11. *Let Γ be a constraint language such that $\mathrm{Pol}(\Gamma) \in \{\mathrm{L}_2, \mathrm{L}_3\}$. Then $\mathrm{CSP}(\Gamma)$ is $\leq_m^{\mathrm{AC}^0}$-hard for $\oplus\mathrm{L}$.*

Proof. Assume without loss of generality that Γ contains $=$. The proof of the general case then follows from Lemmas 2.5 and 3.7, and Proposition 3.6. For the L_2-case, hardness can be shown in a straightforward manner similar to the proof of Lemma 3.10. (We show $\mathrm{SAT}^C(\mathrm{L}_0) \leq_m^{\mathrm{AC}^0} \mathrm{CSP}(\Gamma)$ for a constraint language Γ with $\mathrm{Pol}(\Gamma) = \mathrm{L}_2$. The result then follows with [Rei01]. Since we can express x_{out} and $x_1 = x_2 \oplus x_3$ as L_2-invariant relations, we can directly reproduce the given L_0-circuit.)

This does not work for L_3, since we cannot express x or $\overline{x}$ in L_3. However, since L_3 is basically L_2 plus negation, we can "extend" a given relation from $\mathrm{Inv}(\mathrm{L}_2)$ so that it is invariant under negation, by simply doubling the truth-table.

More precisely, given a constraint language Γ such that $\mathrm{Pol}(\Gamma) = \mathrm{L}_2$, we show that there is a constraint language Γ' such that $\mathrm{Pol}(\Gamma') = \mathrm{L}_3$ and $\mathrm{CSP}(\Gamma) \leq_m^{\mathrm{AC}^0} \mathrm{CSP}(\Gamma')$. For an n-ary relation $R \in \Gamma$, let $\overline{R} = \{(\overline{x_1}, \ldots, \overline{x_n}) \mid (x_1, \ldots, x_n) \in R\}$, and let R' be the $(n+1)$-ary relation $R' = (\{0\} \times R) \cup (\{1\} \times \overline{R})$. It is obvious that R' is closed under N_2 and under L_2, and hence under L_3. Let φ be an instance of $\mathrm{CSP}(\Gamma)$. Let $\Gamma' = \{R' \mid R \in \Gamma\}$. Let $\varphi = \bigwedge_{i=1}^{n} R_n(x_{i_1}, \ldots, x_{i_{n_i}})$. We set $\varphi' = \bigwedge_{i=1}^{n} R'_n(t, x_{i_1}, \ldots, x_{i_{n_i}})$ for a new variable t.

Let $\varphi \in \mathrm{CSP}(\Gamma)$ and $I \models \varphi$. Then $I \cup \{t = 0\} \models \varphi'$.

Let $\varphi' \in \mathrm{CSP}(\Gamma)$ and $I' \models \varphi'$. Without loss of generality, let $I'(t) = 0$ (otherwise, observe $\overline{I'} \models \varphi'$ holds as well), therefore $I'\{t = 0\} \models \varphi$, and thus $\mathrm{CSP}(\Gamma) \leq_m^{\mathrm{AC}^0} \mathrm{CSP}(\Gamma')$ holds. □

With the same technique as in Example 2.4, we can examine the complexity of CSPs invariant under M_2—the constraint satisfaction problems covered in this result are closely related to 2SAT, and hence NL-completeness is a natural result.

Lemma 3.12. *Let Γ be a constraint language such that $\mathrm{Pol}(\Gamma) \subseteq \mathrm{M}_2$. Then $\mathrm{CSP}(\Gamma)$ is $\leq_m^{\mathrm{AC}^0}$-hard for* NL.

Proof. Since $\mathrm{Pol}(\Gamma) \subseteq \mathrm{M}_2$, we know $x \to y$, x, and $\overline{x}$ can be expressed with Γ. Therefore, the graph accessibility problem for directed graphs easily reduces to $\mathrm{CSP}(\Gamma)$: Let G be a directed graph and s, t vertices in G. For every vertex, introduce a variable, and for every edge (v_1, v_2), a constraint $v_1 \to v_2$. Add constraints s and $\overline{t}$. It is clear that the constraint formula is satisfiable if and only if there is no path from s to t in G. Since NL is closed under complement [Imm88], [Sze88], the lemma follows with Lemmas 2.5 and 3.7, and Proposition 3.6. □

4 Quantified Constraint Satisfaction Problems

The problems $\mathrm{CSP}(\Gamma)$ that have been considered in the earlier sections of this paper consist of satisfiable formulas φ that are constructed using relations from Γ. Equivalently, we may consider all of the variables to be existentially quantified, and we are asking if the resulting sentence is true over the Boolean domain $\{0, 1\}$. The *Quantified Constraint Satisfaction Problem* $\mathrm{QCSP}(\Gamma)$ is the corresponding problem, where arbitrary combinations of universal and existential quantifiers are allowed. Thus each problem $\mathrm{QCSP}(\Gamma)$ is a special case of the standard PSPACE-complete problem QBF.

There is a long history of investigations of $\mathrm{QCSP}(\Gamma)$ problems, starting with Schaefer [Sch78], and continuing through the next quarter-century, with papers that eventually established Schaefer's claimed complexity characterization of $\mathrm{QCSP}(\Gamma)$ into polynomial-time solvable and PSPACE-complete [APT79,KKF95,CKS01].

A nice discussion of this history is presented by Chen [Che08], who also presents a unified treatment of the tractable cases of $\mathrm{QCSP}(\Gamma)$.

A variant of $\mathrm{QCSP}(\Gamma)$ is the problem $\mathrm{QCSP}_c(\Gamma)$ where in addition to variables, also the constants 0 and 1 may appear in the clauses. It is easy to see that $\mathrm{QCSP}_c(\Gamma)$ is virtually the same problem as $\mathrm{QCSP}(\Gamma \cup \{\{(0)\},\{(1)\}\})$, where we are allowed to use clauses that force variables to take the Boolean values 0 or 1, respectively. A problem $\mathrm{CSP}_c(\Gamma)$ is defined analogously. The constraint languages Γ that can express these clauses are, (this easily follows from Fact 2.2), exactly those for which $\mathrm{Pol}(\Gamma) \subseteq \mathrm{R}_2$ is true. We thank Edith Hemaspaandra for pointing out that our Theorem 3.1, when combined with the techniques of Chen [Che08], yield the following classification of $\mathrm{QCSP}_c(\Gamma)$.

Theorem 4.1. *Let Γ be a constraint language.finite set of Boolean relations such that $\mathrm{Pol}(\Gamma) \subseteq \mathrm{R}_2$, and let $\Gamma' = \Gamma \cup \{\{(0)\},\{(1)\}\}$. The following classification holds:*

- *If $\mathrm{Pol}(\Gamma') = \mathrm{I}_2$, then $\mathrm{QCSP}_c(\Gamma)$ is $\leq_m^{\mathrm{AC}^0}$-complete for* PSPACE.
- *If $\mathrm{Pol}(\Gamma') \in \{\mathrm{V}_2, \mathrm{E}_2\}$, then $\mathrm{QCSP}_c(\Gamma)$ is $\leq_m^{\mathrm{AC}^0}$-complete for* P.
- *If $\mathrm{Pol}(\Gamma') = \mathrm{L}_2$, then $\mathrm{QCSP}_c(\Gamma)$ is $\leq_m^{\mathrm{AC}^0}$-complete for* $\oplus$L.
- *If $\mathrm{S}_{00} \subseteq \mathrm{Pol}(\Gamma') \subseteq \mathrm{S}_{00}^2$ or $\mathrm{S}_{10} \subseteq \mathrm{Pol}(\Gamma') \subseteq \mathrm{S}_{10}^2$ or $\mathrm{Pol}(\Gamma') \in \{\mathrm{D}_2, \mathrm{M}_2\}$, then $\mathrm{QCSP}_c(\Gamma)$ is $\leq_m^{\mathrm{AC}^0}$-complete for* NL.
- *If $\mathrm{Pol}(\Gamma') = \mathrm{D}_1$, then $\mathrm{QCSP}_c(\Gamma)$ is $\leq_m^{\mathrm{AC}^0}$-complete for* L.
- *If $\mathrm{S}_{02} \subseteq \mathrm{Pol}(\Gamma')$ or $\mathrm{S}_{12} \subseteq \mathrm{Pol}(\Gamma')$, then either $\mathrm{QCSP}_c(\Gamma)$ is in* coNLOGTIME, *or $\mathrm{QCSP}_c(\Gamma)$ is complete for* L *under $\leq_m^{\mathrm{AC}^0}$. There is an algorithm deciding which case occurs.*

Note that since $\mathrm{Pol}(\Gamma') = \mathrm{Pol}(\Gamma) \cap \mathrm{R}_2$ is always a subset of R_2, the above is a complete case distinction. Also, due to the above considerations, in these cases we have that $\mathrm{QCSP}(\Gamma')$ and $\mathrm{QCSP}_c(\Gamma')$ are almost identical problems, in particular, they have the same complexity up to $\leq_m^{\mathrm{AC}^0}$-reductions.

Proof. The PSPACE-hardness results follow via the standard reduction from QBF. For all of the other Γ, $\mathrm{CSP}(\Gamma)$ and therefore (since Γ can express the "constant relations") $\mathrm{CSP}_c(\Gamma)$ is in P. It then follows from Theorems 4.3, 4.6, and 4.9 of Chen [Che08], together with his Proposition 3.12, that $\mathrm{QCSP}_c(\Gamma)$ is reducible in polynomial time to $\mathrm{CSP}_c(\Gamma)$, which can be seen to be the same problem as $\mathrm{CSP}(\Gamma \cup \{\{(0)\},\{(1)\}\})$, which is the same as $\mathrm{CSP}(\Gamma)$, since Γ already can express these relations.

Examination of the proof of [Che08, Proposition 3.12] reveals that this reduction is in fact an $\leq_m^{\mathrm{AC}^0}$ reduction, so that $\mathrm{QCSP}(\Gamma)$ and $\mathrm{CSP}(\Gamma)$ are equivalent under $\leq_m^{\mathrm{AC}^0}$ reductions. The result now follows immediately from Theorem 3.1. □

We remark that, if we restrict the quantifier prefix to have a constant number of alternations, then a similar classification holds, where "PSPACE" is replaced by the appropriate level of the polynomial hierarchy.

Note that a direct variation of the above proof cannot be used to obtain a full classification of QCSP(Γ)—there are cases where CSP(Γ) is solvable in AC^0, but QCSP(Γ) is PSPACE-complete. As an example, consider the case that $Pol(\Gamma) = I_1$, then every Γ-formula is satisfiable by the constant 1-assignment, and hence CSP(Γ) is trivial. However, this knowledge does not help us in deciding whether a formula involving *universal* quantification is true, and in fact, it can be shown that in this case, QCSP(Γ) is PSPACE-complete (see e.g., [CKS01]).

5 Conclusion and Further Research

We have obtained a complete classification for constraint satisfaction problems under AC^0 isomorphisms, and identified six isomorphism types corresponding to the complexity classes $NP, P, NL, \oplus L, L$, and AC^0. One can also show that all constraint satisfaction problems in AC^0 are either trivial or are complete for coNLOGTIME (under logtime-uniform projections).

In a seminal paper, Feder and Vardi [FV99] conjectured that each CSP(Γ) either lies in P or is NP-complete. This so-called dichotomy conjecture is the natural extension to non-Boolean domains of Schaefer's result. Even today, the dichotomy conjecture is only known to hold for domain size two [Sch78] and three [Bul02]. One natural question for further research concerns constraint satisfaction problems over non-Boolean domains. In particular, it would be interesting to see if the dichotomy theorem of Bulatov [Bul02] over three-element domains can be refined to obtain a complete classification up to AC^0-isomorphism. Building on the work presented here, Larose and Tesson [LT07] consider a refinement of the dichotomy conjecture and present algebraic conditions on constraint languages Γ that ensure the hardness of the corresponding constraint satisfaction problem for complexity classes L, NL, Mod_pL, P, and NP.

Acknowledgments

The first and third authors thank Denis Thérien for organizing a workshop at Bellairs research institute where Phokion Kolaitis lectured at length about constraint satisfiability problems. We thank Phokion Kolaitis for his lectures and for stimulating discussions, and in particular, for telling us that it was open whether there are more than five CSPs up to logspace reducibility. The first author thanks Edith Hemaspaandra and Hubie Chen for conversations held at the AIM Workshop on Applications of Universal Algebra and Logic to the Constraint Satisfaction Problem, which led to the results of Section 4. We also thank Nadia Creignou for helpful hints. We thank the anonymous referee for useful suggestions.

References

[ABI97] E. Allender, J. Balcazar, and N. Immerman. A first-order isomorphism theorem. *SIAM Journal on Computing*, 26:557–567, 1997.

[ABI+05] E. Allender, M. Bauland, N. Immerman, H. Schnoor, and H. Vollmer. The complexity of satisfiability problems: refining Schaefer's theorem. In Proc. *Mathematical Foundations of Computer Science (MFCS): 30th International Symposium*, Lecture Notes in Computer Science 3618, pages 71–82, Berlin Heidelberg, 2005. Springer Verlag.

[AG00] C. Alvarez and R. Greenlaw. A compendium of problems complete for symmetric logarithmic space. *Computational Complexity*, 9(2):123–145, 2000.

[Agr01] M. Agrawal. The first-order isomorphism theorem. In Proc. *Foundations of Software Technology and Theoretical Computer Science (FSTTCS): 21st Conference*, Lecture Notes in Computer Science 2245, pages 58–69, Berlin Heidelberg, 2001. Springer Verlag.

[APT79] B. Aspvall, M. Plass, and R. Tarjan. A linear-time algorithm for testing the truth of certain quantified Boolean formulas. *Information Processing Letters*, 8:121–123, 1979.

[At05] Albert Atserias. On Digraph Coloring Problems and Treewidth Duality. In Proc. *20th IEEE Symposium on Logic in Computer Science (LICS)*, pages 106–115, 2005.

[BCRV03] E. Böhler, N. Creignou, S. Reith, and H. Vollmer. Playing with Boolean blocks, part I: Post's lattice with applications to complexity theory. *SIGACT News*, 34(4):38–52, 2003.

[BCRV04] E. Böhler, N. Creignou, S. Reith, and H. Vollmer. Playing with Boolean blocks, part II: Constraint satisfaction problems. *SIGACT News*, 35(1):22–35, 2004.

[BRSV05] E. Böhler, S. Reith, H. Schnoor, and H. Vollmer. Simple bases forBoolean co-clones. *Information Processing Letters*, 96:59–66, 2005.

[Bul02] A. Bulatov. A dichotomy theorem for constraints on a three-element set. In *Proceedings 43rd Symposium on Foundations of Computer Science*, pages 649–658. IEEE Computer Society Press, 2002.

[Che08] H. Chen. The Complexity of Quantified Constraint Satisfaction: Collapsibility, Sink Algebras, and the Three-Element Case. *SIAM Journal on Computing*, 37:1674–1701, 2008.

[CJ06] D.A. Cohen and P.G. Jeavons. The complexity of constraint languages. In *Handbook of Constraint Programming*, Elsevier, 2006.

[CKS01] N. Creignou, S. Khanna, and M. Sudan. *Complexity Classifications of Boolean Constraint Satisfaction Problems*. Monographs on Discrete Applied Mathematics. SIAM, 2001.

[Dal05] V. Dalmau. Linear Datalog and bounded path duality of relational structures. *Logical Methods in Computer Science*, 1(1), 2005.

[FV99] T. Feder and M. Y. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: A study through Datalog and group theory. *SIAM J. on Computing*, 28(1):57–104, 1999.

[Gei68] D. Geiger. Closed systems of functions and predicates. *Pac. J. Math*, 27(1):95–100, 1968.

[HO02] L. Hemaspaandra and M. Ogihara. *The complexity theory companion*. Springer-Verlag New York, Inc., New York, NY, USA, 2002.

[Imm88] N. Immerman. Nondeterministic space is closed under complementation. *SIAM Journal on Computing*, 17:935–938, 1988.

[Imm99] N. Immerman. *Descriptive Complexity*. Springer Graduate Texts in Computer Science, Springer-Verlag New York, Inc., New York, NY, USA, 1999.

[JCG97] P. G. Jeavons, D. A. Cohen, and M. Gyssens. Closure properties of constraints. *Journal of the ACM*, 44(4):527–548, 1997.

[KKF95] H. Kleine Büning, M. Karpinski, and A. Flögel. Resolution for quantified Boolean formulas. *Information and Computation*, 117:12–18, 1995.

[LLT07] B. Larose, C. Loten, and C. Tardif. A Characterisation of first-order constraint satisfaction problems. *Logical Methods in Computer Science* 3(4), 2007.

[LT07] B. Larose and P. Tesson. Universal algebra and hardness results for constraint satisfaction problems. In Proc. *Automata, Languages and Programming: 34th International Colloquium (ICALP)*, Lecture Notes in Computer Science 4596, pages 267–278, Berlin Heidelberg, 2007. Springer Verlag.

[Lau06] Lau, D., 2006. *Function Algebras on Finite Sets: Basic Course on Many-Valued Logic and Clone Theory (Springer Monographs in Mathematics)*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

[Pip97] N. Pippenger. *Theories of Computability*. Cambridge University Press, Cambridge, 1997.

[Pos20] E. L. Post. Determination of all closed systems of truth tables. *Bulletin of the AMS*, 26:437, 1920.

[Pos41] E. L. Post. The two-valued iterative systems of mathematical logic. *Annals of Mathematical Studies*, 5:1–122, 1941.

[Rei01] S. Reith. *Generalized Satisfiability Problems*. PhD thesis, Fachbereich Mathematik und Informatik, Universität Würzburg, 2001.

[Rei05] O. Reingold. Undirected st-connectivity in log-space. In *Proceedings 37th Symposium on Foundations of Computer Science*, pages 376–385. IEEE Computer Society Press, 2005.

[Ros05] B. Rossman. Existential Positive Types and Preservation under Homomorphisisms. In Proc. *20th IEEE Symposium on Logic in Computer Science (LICS)*, pages 467–476, 2005.

[Sch78] T. J. Schaefer. The complexity of satisfiability problems. In *Proceedings 10th Symposium on Theory of Computing*, pages 216–226. ACM Press, 1978.

[Sze88] R. Szelepcsényi. The method of forced enumeration for nondeterministic automata. *Acta Informatica*, 26:279–284, 1988.

[Vol99] H. Vollmer. *Introduction to Circuit Complexity – A Uniform Approach*. Texts in Theoretical Computer Science. Springer Verlag, Berlin Heidelberg, 1999.