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

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

[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|schnoor|vollmer@thi.uni-hannover.de

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

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).

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 has 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]).

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

* Supported in part by DFG grant Vo 630/5-1.

study of Boolean constraint satisfaction problems that does *not* appear in the list $(\mathrm{AC}^0, \mathrm{NL}, \oplus\mathrm{L}, \mathrm{P})$ of 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 $\mathrm{NP}, \mathrm{P}, \oplus\mathrm{L}, \mathrm{NL}$, or L under isomorphisms computable and invertible in AC^0.

Our proofs rely heavily on the connection between complexity of constraint languages and universal algebra (in particular, the theory of *polymorphisms* and *clones*) which has been very useful in analyzing complexity issues of constraints. An introduction to this connection can be found in [Pip97b], and we recall some of the necessary definitions in the next section. 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

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 $R(x_1, \ldots, x_n)$ iff $(I(x_1), \ldots, I(x_n)) \in R$. In this paper we use the standard correspondence between Boolean relations and propositional formulas: A formula $\varphi(x_1, \ldots, x_n)$ defines the relation $R_\varphi = \{(\alpha_1, \ldots, \alpha_n) \mid \varphi(\alpha_1, \ldots, \alpha_n) = 1\}$. The meaning should always be clear from the context.

A *constraint language* is a finite set of Boolean relations. The Boolean *Constraint Satisfaction Problem* over a constraint language Γ ($\mathrm{CSP}(\Gamma)$) is the question if 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. A well-known restriction of the general satisfiability problem is 3SAT, which can be seen as the CSP problem over the language $\Gamma_{3\mathrm{SAT}} = \{(x_1 \vee x_2 \vee x_3), (\overline{x_1} \vee x_2 \vee x_3), (\overline{x_1} \vee \overline{x_2} \vee x_3), (\overline{x_1} \vee \overline{x_2} \vee \overline{x_3})\}$.

There is a very useful connection between the complexity of the CSP problem and universal algebra, which requires a few definitions:

A class of Boolean functions is called *closed* or a *clone*, if it is closed under superposition. (As explained in the surveys [BCRV03,BCRV04] being closed under superposition is essentially the same thing as containing all projections (in particular, the identity) and being closed under arbitrary composition.) Since

the intersection of clones is again a clone, we can define, for a set B of Boolean functions, $\langle B \rangle$ as the smallest clone containing B.

It is clear that $\langle B \rangle$ is the set of Boolean functions that can be calculated by Boolean circuits using only gates for functions from B [BCRV03,Pip97a].

It is easy to see that the set of clones forms a lattice. For the Boolean case, Emil Post 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. The clones are interesting for the study of the complexity of CSPs, because the complexity of CSP(Γ) depends on the closure properties of the relations in Γ, which we will define next.

Definition 2.1. *A k-ary relation R is* closed *under an n-ary Boolean function f, or f is a* polymorphism *of R, 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.*

It is easy to see that every set of the form Pol(Γ) is a clone. The operators Pol and Inv form a "Galois connection" between the lattice of clones and certain sets of Boolean relations, which is very useful for complexity analysis of the CSP problem. 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. A proof of the following theorem can be found in e.g. [JCG97] and [Dal00]:

Theorem 2.2. *If* $\mathrm{Pol}(\Gamma_2) \subseteq \mathrm{Pol}(\Gamma_1)$*, then every $R \in \Gamma_1$ can be expressed by a formula*

$$R(x_1, \ldots, x_n) \iff \exists y_1, \ldots, y_m R_1(z_{1,1}, \ldots, z_{1,n_1}) \wedge \cdots \wedge R_k(z_{k,1}, \ldots, z_{k,n_k})$$
$$\wedge (x_{i_1} = x_{i_2}) \wedge (x_{i_3} = x_{i_4}) \wedge \cdots \wedge (x_{i_{r-1}} = x_{i_r})$$

for some $R_i \in \Gamma_2$ (where $z_{i,j} \in \{x_1, \ldots, x_n, y_1, \ldots, y_m\}$).

Therefore:

Theorem 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* $\mathrm{CSP}(\Gamma_1) \leq_m^{\mathrm{p}} \mathrm{CSP}(\Gamma_2)$*.*

Name	Definition	Base
BF	All Boolean functions	$\{\vee, \wedge, \neg\}$
R_0	$\{f \in \mathrm{BF} \mid f$ is 0-reproducing $\}$	$\{\wedge, \oplus\}$
R_1	$\{f \in \mathrm{BF} \mid f$ is 1-reproducing $\}$	$\{\vee, \leftrightarrow\}$
R_2	$R_1 \cap R_0$	$\{\vee, x \wedge (y \leftrightarrow z)\}$
M	$\{f \in \mathrm{BF} \mid f$ 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	$\{f \in \mathrm{BF} \mid f$ is 0-separating of degree $n\}$	$\{\rightarrow, \mathrm{dual}(h_n)\}$
S_0	$\{f \in \mathrm{BF} \mid f$ is 0-separating $\}$	$\{\rightarrow\}$
S_1^n	$\{f \in \mathrm{BF} \mid f$ is 1-separating of degree $n\}$	$\{x \wedge \overline{y}, h_n\}$
S_1	$\{f \in \mathrm{BF} \mid f$ is 1-separating $\}$	$\{x \wedge \overline{y}\}$
S_{02}^n	$S_0^n \cap R_2$	$\{x \vee (y \wedge \overline{z}), \mathrm{dual}(h_n)\}$
S_{02}	$S_0 \cap R_2$	$\{x \vee (y \wedge \overline{z})\}$
S_{01}^n	$S_0^n \cap M$	$\{\mathrm{dual}(h_n), 1\}$
S_{01}	$S_0 \cap M$	$\{x \vee (y \wedge z), 1\}$
S_{00}^n	$S_0^n \cap R_2 \cap M$	$\{x \vee (y \wedge z), \mathrm{dual}(h_n)\}$
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$\}$	$\{x\overline{y} \vee x\overline{z} \vee (\overline{y} \wedge \overline{z})\}$
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$\}$	
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$\}$	
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	$[\{\mathrm{id}\}] \cup [\{0\}] \cup [\{1\}]$	$\{\mathrm{id}, 0, 1\}$
I_0	$[\{\mathrm{id}\}] \cup [\{0\}]$	$\{\mathrm{id}, 0\}$
I_1	$[\{\mathrm{id}\}] \cup [\{1\}]$	$\{\mathrm{id}, 1\}$
I_2	$[\{\mathrm{id}\}]$	$\{\mathrm{id}\}$

Table 1: List of all closed classes of Boolean functions, and their bases (for definitions of these properties, see e.g. [BCRV03]).

The function h_n is defined as:

$$h_n(x_1, \ldots, x_{n+1}) = \bigvee_{i=1}^{n+1} x_1 \wedge x_2 \wedge \cdots \wedge x_{i-1} \wedge x_{i+1} \wedge \cdots \wedge x_{n+1}$$

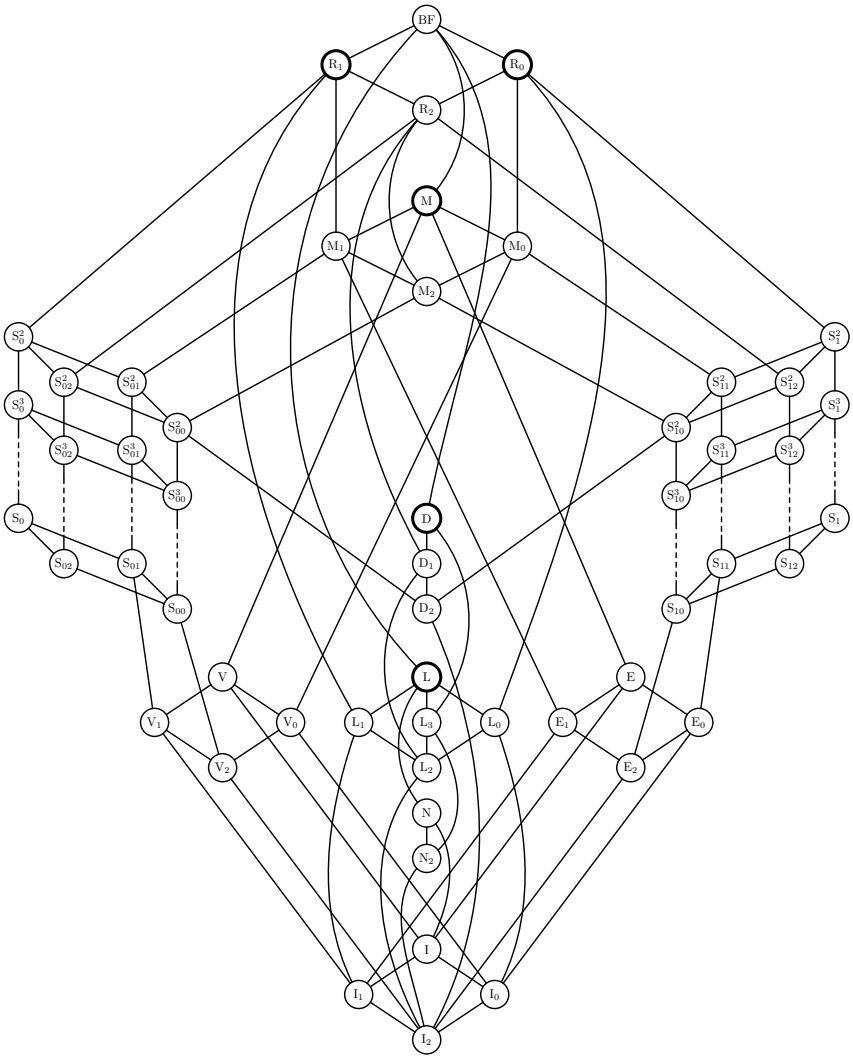

Fig. 1. Graph of all closed classes of Boolean functions

Trivially, the binary equality predicate $=$ is closed under every Boolean function. Thus, $=$ is contained in every set $\mathrm{Inv}(B)$ for a clone B (these sets often are called *co-clones*). On the other hand, every relation is closed under the projection function, $\phi_i^n(x_1, \ldots, x_n) = x_i$. It is clear that when a set of relations is "big", the set of its polymorphisms is "small". So the most general case is a constraint language Γ such that $\mathrm{Pol}(\Gamma)$ only contains the projections, and these cases of the CSP are NP-complete. An example for this is the language $\Gamma_{3\mathrm{SAT}}$ from above: It can be shown that $\mathrm{Pol}(\Gamma_{3\mathrm{SAT}})$ only contains the projections, and therefore 3SAT is NP-complete.

As we have seen in the above theorem, the complexity of the CSP problem 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 Theorem 2.2 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

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 the standard complete set now follows by [ABI97] (since the standard complete set is complete under invertible, length-squaring reductions). □

3.1 Upper Bounds: Algorithms

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

Proposition 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* $\mathrm{CSP}(\Gamma) \in \oplus\mathrm{L}$.
3. $\mathrm{D}_2 \subseteq \mathrm{Pol}(\Gamma)$ *implies* $\mathrm{CSP}(\Gamma) \in \mathrm{NL}$.
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.

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 $\mathrm{Pol}(\Gamma) = \mathrm{D}$.

This can easily be shown by introducing a new variable f and replacing clauses x with $x \oplus f$ (observe that $\oplus$ is closed under negation). □

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 Theorem 2.2 we can construct "$x \to y$" with 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 Theorem 2.2). 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 $R^4_{\mathrm{even}}(z, z, x, y)$. If $\mathrm{Pol}(\Gamma) \subseteq \mathrm{D}$, 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 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 $\mathrm{Pol}(\Gamma) \supseteq \mathrm{S}^m_{02}$ or $\mathrm{Pol}(\Gamma) \supseteq \mathrm{S}^m_{12}$ holds for some $m \geq 2$.

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

Proof. If $\mathrm{Pol}(R) \supseteq \mathrm{S}_{02}$, then as in the proof of Lemma 3.4 we know that $\mathrm{Pol}(R) \supseteq \mathrm{S}^m_{02}$ for some $m \geq 2$. Thus we know from Theorem 2.2 that R can be expressed using equality, literals, and the m-ary OR predicate, since $\mathrm{Pol}(\{x, \overline{x}, \mathrm{OR}^m\}) = \mathrm{S}^m_{02}$ ([BRSV05]). 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):

1. For any clause $x_1 = x_2$ where x_1 or x_2 appears as a literal, remove this clause and insert the corresponding literals for x_1 and x_2. 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 $=$, remove all of them except one.

Note that this does not change the relation represented by the formula. 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 iff 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 that claim 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, this is a finite number of combinations which can be easily verified by an algorithm. An analogous argument can be applied to the dual case $\mathrm{Pol}(R) \supseteq S_{12}^m$. □

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 3.9. *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. It is well-known that the satisfiability problem for Horn and anti-Horn formulas is complete for P. To show that this hardness results holds under AC^0

reductions, it is easy to construct a reduction from $\mathrm{SAT}^C(\mathrm{S}_{11})$, by just simulating every gate in the circuit as an anti-Horn clause. The result then follows from [Rei01]. The dual case for Horn-formulas is analogous. □

Lemma 3.10. *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 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, 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 $\varphi \in \mathrm{CSP}(\Gamma)$, $I \models \varphi$. Then $I \cup \{t = 0\} \models \varphi'$.

Let $\varphi' \in \mathrm{CSP}(\Gamma)$, $I' \models \varphi'$. 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 relation $x \to y$ is invariant under M_2, and thus we can model search in directed graphs here.

Lemma 3.11. *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.

4 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 $\mathrm{NP}, \mathrm{P}, \mathrm{NL}, \oplus\mathrm{L}, \mathrm{L}$, 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).

One natural question for further research concerns constraint satisfaction problems over larger 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.

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. We also thank Nadia Creignou for helpful hints.

References

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

[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 *Foundations of Software Technology and Theoretical Computer Science: 21st Conference, Bangalore, India, December 13-15, 2001. Proceedings*, Lecture Notes in Computer Science, pages 58–69, Berlin Heidelberg, 2001. Springer Verlag.

[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 for Boolean co-clones. *Information Processing Letters*, 2005. to appear.

[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.

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

[Dal00] V. Dalmau. *Computational complexity of problems over generalized formulas*. PhD thesis, Department de Llenguatges i Sistemes Informàtica, Universitat Politécnica de Catalunya, 2000.

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

[Pip97a] N. Pippenger. Pure versus impure Lisp. *ACM Transactions on Programming Languages and Systems*, 19:223–238, 1997.

[Pip97b] N. Pippenger. *Theories of Computability*. Cambridge University Press, Cambridge, 1997.

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

[Rei05] Omer Reingold. Undirected st-connectivity in log-space. In *STOC '05: Proceedings of the thirty-seventh annual ACM symposium on Theory of computing*, pages 376–385, New York, NY, USA, 2005. ACM Press.

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