The Complexity of Equality Constraint Languages

Manuel Bodirsky[1] and Jan Kára[2]

[1] Department Algorithms and Complexity, Humboldt University, Berlin, bodirsky@informatik.hu-berlin.de

[2] Department of Applied Mathematics, Faculty of Mathematics and Physics, Charles University, Prague, kara@kam.mff.cuni.cz ***

Abstract. We apply the algebraic approach to infinite-valued constraint satisfaction to classify the computational complexity of all constraint languages where the constraint types are Boolean combinations of the equality relation. We show that such a constraint language is tractable if it admits a constant unary or an injective binary polymorphism, and is NP-complete otherwise.

1 Introduction

In a constraint satisfaction problem we are given a set of variables and a set of constraints on that variables, and want to find an assignment of values to the variables such that all the constraints are satisfied. We are interested in the computational complexity of the constraint satisfaction problem depending on the constraint language that we are allowed to use in the instances of the constraint satisfaction problem; see e.g. [6] for an introduction to the state-of-the-art of the techniques used to study the computational complexity of constraint satisfaction problems.

Formally, we can define constraint satisfaction problems (CSPs) as *homomorphism problems* for relational structures. Let Γ be a (not necessarily finite) structure with a relational signature τ. Then the constraint satisfaction problem CSP(Γ) is a computational problem, where we are given a *finite* τ-structure S and want to know whether there is a homomorphism from S to Γ; for the detailed definitions, see Section 2. It is easy to see that the class of constraint satisfaction problems equals the class of problems that is closed under so-called *inverse homomorphisms* (if we add constraints to an unsatisfiable instance it stays unsatisfiable) and *disjoint unions* (two satisfiable constraints on distinct variables have a joint solution). We show two examples.

*** The author has been supported by a Marie Curie Fellowship of the European Community programme "Combinatorics, Geometry, and Computation" under contract number HPMT-CT-2001-00282.

Example 1. Let Γ be the relational structure $(\mathbb{N}; =, \neq)$. Then $\mathrm{CSP}(\Gamma)$ is the computational problem to decide for a given set of equalities and inequalities on a finite set of variables whether the variables can be mapped to the natural numbers such that variables with a constraint $x = y$ are mapped to the same, and variables with a constraint $x \neq y$ are mapped to distinct values.

This problem is tractable: we sketch a simple algorithm that solves this problem in polynomial time. The algorithm iteratively identifies variables with an equality constraint. If it has to identify two variables with an inequality constraint, it outputs that the constraint has no solution. Otherwise, we know that we can finally map all the remaining variables to distinct values and satisfy all the constraints.

Example 2. Let Γ be the relational structure $(\mathbb{N}; \neq, Q)$, where Q is the relation $Q := \{(x, y, z) \in \mathbb{N}^3 \mid x = y \vee y = z\}$. Here the problem $\mathrm{CSP}(\Gamma)$ turns out to be NP-complete (see Section 3).

In this paper we consider constraint satisfaction problems with templates of the form $\Gamma = (V; R_1, \ldots, R_k)$ where V is a countably infinite domain and each relation $R_1, \ldots, R_k$ is a Boolean combination of atoms of the form $x = y$. (A Boolean combination is a relation built from atomic relations with the usual operations of intersection, union, and complementation.) We say that such a relational structure defines an *equality constraint language*. Later, we also discuss the case where the template has infinitely many relation symbols $R_1, R_2, \ldots$. Note that Example 1 and 2 are both equality constraint languages. The main result of this paper is the following (again, for definitions of the involved concepts see Section 2).

Theorem 1. *An equality constraint language with template Γ is tractable if Γ has a constant endomorphism or an injective homomorphism from Γ^2 to Γ. Otherwise it is NP-complete.*

In Theorem 1, the containment in NP is easy to see: a nondeterministic algorithm can guess which variables in an instance S denote the same element in Γ, and can verify whether this gives rise to a solution for S. Both the hardness result and the algorithmic tractability result in Theorem 1 are nontrivial. The hardness proof presented in Section 3 relies on the algebraic approach to constraint satisfaction, which was previously mainly applied to constraint satisfaction with finite templates.

The algorithm introduced in Section 4 that solves all problems with an injective binary polymorphism is of a new type, as compared to the known algorithms that are used to solve tractable constraint satisfaction problems with finite templates. Note that the 'simple' algorithm that was described in the beginning and solves Problem 1 does not work in general for problems that are closed under an injective binary polymorphism. Consider for example the problem $\mathrm{CSP}(\mathbb{N}; R)$, where R is the 4-ary relation defined as follows.

$$R := \{(x_1, y_1, x_2, x_3) \mid (x_1 = y_1 \wedge x_1 \neq x_2 \neq x_3 \neq x_1) \vee x_1 = y_1 = x_2 = x_3\}$$

Let S be the following instance of this problem on five variables denoted by a, b, c, d, e, where the constraint R is imposed on (a, b, c, d) and on (b, c, d, e). These constraints induce the equalities $a = b = c$. But assigning different values to each equivalence class is not a correct assignment for S, because the first constraint is violated. The only correct assignment here is to assign the same value to all the variables.

The algebraic approach. A formula is called *primitive positive*, if it has the form $\exists x_1 \dots x_k . \psi_1 \wedge \dots \wedge \psi_l$, where ψ_i is an atomic formula that might contain free variables and existentially quantified variables from $x_1, \dots, x_k$. A formula is called *existential positive*, if it is a disjunctive combination of primitive positive formulas (equivalently, if it is a first-order formula without universal quantifiers and negations). Every formula with k free variables defines on a structure Γ a k-ary relation. Primitive positive definability of relations is an important concept in constraint satisfaction, because primitive positive definable relations can be 'simulated' by the constraint satisfaction problem. The following is frequently used in hardness proofs for constraint satisfaction problems; see e.g. [11].

Lemma 1. *Let Γ be a relational structure, and let R be a relation that has a primitive positive definition in Γ. Then the constraint satisfaction problems of Γ and of the expansion of Γ by R have the same computational complexity.*

The algebraic approach to constraint satisfaction (see e.g. [4, 5, 11]) is based on the following preservation statements that characterize syntactic restrictions of first-order definability. For a formal definitions of all the involved concepts, see Section 2.

Theorem 2 (from [3, 8, 12]). *Let Γ be a finite relational structure. Then*

1. *A relation R has a first-order definition in Γ if and only if it is preserved by all automorphisms of Γ;*
2. *A relation R has an existential positive definition in Γ if and only if it is preserved by all endomorphisms of Γ;*
3. *A relation R has a primitive positive definition in Γ if and only if it is preserved by all homomorphisms from Γ^k to Γ, for all $k \geq 1$.*

These statements do not hold for infinite structures in general. However, we have the following.

Theorem 3 (from [1, 2]). *Let Γ be a countably infinite relational structure. Then Statement 1 of Theorem 2 holds if and only if Γ is ω-*categorical*, i.e., if the first-order theory of Γ has only one countable model up to isomorphism. For ω-categorical Γ, Statements 2 and 3 hold as well.*

The templates for equality constraint languages are easily seen to be ω-categorical, since they all have a first-order definition in the ω-categorical empty structure on a countable set of vertices; see e.g. [10].

2 Fundamental Concepts for the Algebraic Approach

We introduce classical concepts that are fundamental for the algebraic approach to constraint satisfaction.

Structures. A *relational signature* τ is a (here always at most countable) set of *relation symbols* R_i, each associated with a finite *arity* k_i. A *(relational) structure* Γ *over relational signature* τ (also called τ*-structure*) is a countable set D_Γ (the *domain*) together with a relation $R_i \subseteq D_\Gamma^{k_i}$ for each relation symbol of arity k_i. For simplicity, we use the same symbol for a relation symbol and the corresponding relation. If necessary, we write R^Γ to indicate that we are talking about the relation R belonging to the structure Γ. For a τ-structure Γ and $R \in \tau$ it will also be convenient to say that $R(u_1, \ldots, u_k)$ *holds in* Γ iff $(u_1, \ldots, u_k) \in R$. If we add relations to a given structure Γ we call the resulting structure Γ' an *expansion* of Γ, and Γ is called a *reduct* of Γ'.

Homomorphisms. Let Γ and Γ' be τ-structures. A *homomorphism* from Γ to Γ' is a function f from D_Γ to $D_{\Gamma'}$ such that for each n-ary relation symbol R in τ and each n-tuple $(a_1, \ldots, a_n)$, if $(a_1, \ldots, a_n) \in R^\Gamma$, then $(f(a_1), \ldots, f(a_n)) \in R^{\Gamma'}$. In this case we say that the map f *preserves* the relation R. Isomorphisms from Γ to Γ are called *automorphisms*, and homomorphisms from Γ to Γ are called *endomorphisms*. The set of all automorphisms of a structure Γ is a group, and the set of all endomorphisms of a structure Γ is a monoid with respect to composition.

Polymorphisms. Let D be a countable set, and O be the set of *finitary operations* on D, i.e., functions from D^k to D for finite k. We say that a k-ary operation $f \in O$ *preserves* an m-ary relation $R \subseteq D^m$ if whenever $R(x_1^i, \ldots, x_m^i)$ holds for all $1 \leq i \leq k$ in Γ, then $R\big(f(x_1^1, \ldots, x_1^k), \ldots, f(x_m^1, \ldots, x_m^k)\big)$ holds in Γ. If f preserves all relations of a relational τ-structure Γ, we say that f is a polymorphism of Γ. In other words, f is a homomorphism from $\Gamma^k = \Gamma \times \ldots \times \Gamma$ to Γ, where $\Gamma_1 \times \Gamma_2$ is the *(categorical- or cross-) product* of the two relational τ-structures Γ_1 and Γ_2. Hence, the unary polymorphisms of Γ are the endomorphisms of Γ, and the unary bijective polymorphisms are the automorphisms of Γ.

Clones. An operation π is a *projection* if for all n-tuples, $\pi(x_1, \ldots, x_n) = x_i$ for some fixed $i \in \{1, \ldots, n\}$. The *composition* of a k-ary operation f and k operations $g_1, \ldots, g_k$ of arity n is an n-ary operation defined by

$$f(g_1, \ldots, g_k)(x_1, \ldots, x_n) = f\big(g_1(x_1, \ldots, x_n), \ldots, g_k(x_1, \ldots, x_n)\big) \quad .$$

A *clone* F is a set of operations from O that is closed under compositions and that contains all projections. We write D_F for the *domain* D of the clone F. It is easy to verify that the set $Pol(\Gamma)$ of all polymorphisms of Γ is a clone with domain D_Γ. Moreover, $Pol(\Gamma)$ is also closed under interpolations: We say that an operation $f \in O$ is *interpolated* by a set $F \subseteq O$ if for every finite subset B of

D there is some operation $g \in F$ such that $f|_B = g|_B$ (f restricted to B equals g restricted to B, i.e., $f(t) = g(t)$ for every $t \in B^k$). The set of operations that are interpolated by F is called the *local closure* of F; if F equals its local closure, we say that F is *locally closed*. For a set of operations F from O we write $\langle F \rangle$ for the smallest locally closed clone containing all operations in F (the clone *locally generated* by F). The following is a well-known fact:

Proposition 1 (see e.g. [15]). *A set $F \subseteq O$ of operations is locally closed if and only if F is the set of polymorphisms of Γ for some relational structure Γ.*

Oligomorphic clones. A permutation group G on a countably infinite set D is called *oligomorphic*, if it has only finitely many orbits of G in its natural action on n-tuples from D, for all $n \geq 1$; see [7]. Similarly, we now define oligomorphic clones.

Definition 1. *A clone on a countably infinite set is called* oligomorphic*, if its bijective unary operations form an oligomorphic permutation group.*

Definition 2. *A countable relational structure Γ is called* ω-categorical*, if all countable models of the first-order theory of Γ are isomorphic to Γ.*

The next theorem can be seen as a reformulation of the theorem of Ryll-Nardzewski, Engeler, and Svenonius (see [10]), and is also equivalent to the first part of Theorem 3.

Theorem 4 (See [7]). *Let Γ be a relational structure. Then the following are equivalent.*

- *Γ is ω-categorical;*
- *the set of polymorphisms of Γ forms an oligomorphic clone;*
- *every k-ary first-order definable relation in Γ is the union of a finite number of orbits of k-tuples of the automorphism group of Γ.*

3 A Generic Hardness Proof

We return to the class of equality constraint languages, which is a subclass of the satisfaction problems with ω-categorical templates. Throughout this section, Γ is a relational structure $(V; R_1, R_2, \dots)$ on a countably infinite domain V where every relation R_i is a Boolean combination of atoms of the form $x = y$. Note that the automorphism group of Γ is the full symmetric group on V, which is clearly oligomorphic.

Lemma 2. *If Γ has a non-injective endomorphism f, then Γ also has a constant endomorphism.*

Proof. Let f be an endomorphism of Γ such that $f(x) = f(y)$ for two distinct points x, y from V. Let $a_1, a_2, \ldots$ be an enumeration of V. We construct an infinite sequence of endomorphisms $e_1, e_2, \ldots$ where e_i is an endomorphism that maps the points $a_1, \ldots, a_i$ to a_1, and maps all other points to values different from a_1. This suffices, since by local closure the mapping defined by $e(x) = a_1$ for all x is an endomorphism of Γ.

For e_1 we take the identity map, which clearly is an endomorphism with the desired properties. To define e_i for $i \geq 2$ let h be an automorphism of Γ that maps $a_1 = e_{i-1}(a_1) = \cdots = e_{i-1}(a_{i-1})$ to x, and $e_{i-1}(a_i)$ to y. Then the endomorphism $f(h(e_{i-1}))$ is constant on $a_1, \ldots, a_i$. There is also an automorphism h' that maps $f(h(e_{i-1}(a_1)))$ to a_1. Then $e_i := h'(f(h(e_{i-1})))$ is an endomorphism with the desired properties. $\square$

Lemma 2 holds in general for ω-categorical structures Γ that have only one orbit of 2-element subsets in the automorphism group of Γ. Similarly, the following holds for ω-categorical structures Γ with only one orbit of pairs of distinct elements in the automorphism group of Γ.

Lemma 3. *If Γ does not have a constant endomorphism, then there is a primitive positive definition of the relation $x \neq y$ in Γ.*

Proof. If Γ does not have a constant endomorphism, Lemma 2 implies that all endomorphisms preserve the inequality relation. By Theorem 3, inequality has an existential positive definition. Since the inequality relation is formed by a single orbit of the automorphism group on pairs of elements (and each primitive positive term adds at least one orbit on tuples), we do not need disjunctions in its existential positive definition, and therefore the inequality relation even has a primitive positive definition. $\square$

Lemma 4. *If the relation $S := \{(x, y, z) \mid x = y \neq z \vee x \neq y = z\}$ has a primitive positive definition in Γ, then CSP(Γ) is NP-hard.*

Proof. First observe that by identification of arguments x and y, if S has a primitive positive definition in Γ, then the inequality relation has a primitive positive definition in Γ as well. We prove the NP-hardness by reduction from the NP-hard problem 3-COLORING. Let $G = (V, E)$ be a graph that is an instance of 3-COLORING. We construct an instance of CSP(Γ) that has a polynomial size in $|V|$ and $|E|$ and is satisfiable if and only if G has a proper 3-coloring. Lemma 1 asserts we can use inequality constraints and the relation S to formulate this instance. The set of variables in this instance is $V \cup V' \cup \{c_1, c_2, c_3\}$, where V' is a copy of V, and c_1, c_2, c_3 are three new variables representing colors. We impose inequality constraints on each pair in c_1, c_2, c_3 and on each pair (u, v) for $uv \in E$. We impose the constraint S on (c_1, v', c_2) for each $v' \in V'$, and on (v', v, c_3) for each $v \in V$ where v' is the copy of v in V'. It is now straightforward to check that in every solution to these constraints each $v \in V$ equals one of the values for c_1, c_2, c_3, and hence solutions of the constructed problem one-to-one correspond to the proper 3-colorings of G. $\square$

Theorem 5. *If Γ has no constant unary and no injective binary polymorphism, then CSP(Γ) is NP-complete.*

Proof. Assume that Γ has no injective binary and no constant unary polymorphism. We claim that in this case every polymorphism is essentially unary. Suppose there is an at least binary polymorphism f that depends on at least two of its arguments, say on the first and the second argument. Then the operation $g(x, y) := f(x, y, \ldots, y)$ is a binary polymorphism, which cannot be injective by assumption, and hence there are two distinct pairs t and t' such that $g(t) = g(t')$. We show that the operation g does not depend on the arguments where t and t' differ, a contradiction.

Let s and s' be two pairs where s and s' differ in the arguments where t and t' differ. We have to show that the difference does not affect the value of g, i.e., that $g(s) = g(s')$. For that, note that for $i = 1, 2$ there is an automorphism h_i that maps $s(i)$ to $t(i)$ and $s'(i)$ to $t'(i)$. Then $g(s) = g(s')$ iff $g(h_1(s(1)), h_2(s(2)))) = g(h_1(s'(1)), h_2(s'(2))))$ iff $g(t) = g(t')$, which holds by assumption. Hence, every polymorphism is essentially unary.

If there is no constant unary polymorphism, then Lemma 2 asserts that every essentially unary operation is injective, and therefore in particular preserves the relation S. Therefore *all* polymorphisms preserve this relation, and by Theorem 3 it has a primitive positive definition. Lemma 4 implies that CSP(Γ) is NP-hard. □

4 Tractability Results

The case that Γ contains a constant unary polymorphism gives rise to trivially tractable constraint satisfaction problems: If an instance of such a constraint satisfaction problem has a solution, then there is also a solution that maps all variables to a single point. In this case an instance of CSP(Γ) is satisfiable if and only if it does not contain a constraint R where R denotes the empty relation in Γ. Clearly, this can be tested efficiently. To finish the classification of the complexity of equality constraint languages we are left with the case that Γ has a binary injective polymorphism.

The algorithm that we are going to present uses a special representation of the relations in Γ. Theorem 4 implies that every k-ary relation in Γ is a union of orbits of k-tuples of the automorphism group of Γ. Let t be a k-tuple from one of these orbits. We define the equivalence relation ρ on the set $\{1, \ldots, k\}$ that contains those pairs $\{i, j\}$ where $t(i) = t(j)$. Clearly, all tuples in the orbit lead to the same equivalence relation ρ. Hence, every k-ary relation R in Γ corresponds uniquely to a set of equivalence relations on $\{1, \ldots, k\}$, which we call the *representation* of R. Sometimes we identify a relation R from Γ with its representation, and for example freely write $\rho \in R$ if ρ is an equivalence relation from the representation of R. Let $|R|$ denote the number of orbits of k-tuples contained in R. Hence, $|R|$ also denotes the number of equivalence relations in the representation of R. Next, we define several notions for equivalence relations that will be useful to formulate our algorithm.

Definition 3. *Let ρ and ρ' be equivalence relations on a set X. We say that ρ is* finer *than ρ', and write $\rho \subseteq \rho'$, if $\rho(x, y)$ implies $\rho'(x, y)$ for each $x, y \in X$. We also say that ρ' is* coarser *than ρ. The* intersection *of these two equivalence relations, denoted by $\rho \cap \rho'$, is the equivalence relation σ such that $\sigma(x, y)$ if and only if $\rho(x, y)$ and $\rho'(x, y)$.*

Lemma 5. *If Γ has a binary injective polymorphism, then for every relation R from Γ the corresponding set of equivalence relations is* closed under intersections, *i.e., $\rho \cap \rho' \in R$ for all equivalence relations $\rho, \rho' \in R$.*

Proof. Let R be a k-ary relation in Γ, and let ρ and ρ' be two equivalence relations from the representation of R. Pick two k-tuples t and t' in R that lie in the orbits that are described by ρ and ρ'. If f is the injective binary polymorphism of Γ, then by injectivity of f the k-tuple $t'' := (f(t(1), t'(1)), \ldots, f(t(k), t'(k)))$ satisfies $t''(i) = t''(j)$ if and only if $\rho(i, j)$ and $\rho'(i, j)$. Hence we found a tuple in R that lies in the orbit that is described by $\rho \cap \rho'$, which is therefore also contained in the representation of R. $\square$

We would like to remark that if the conclusion of Lemma 5 holds for all relations R is Γ, then Γ has a binary injective polymorphism. In fact, it is easy to verify that in this case *any* injection from Γ^2 to Γ preserves all relations.

Lemma 6. *Let Γ be closed under a binary injective polymorphism, and let R be a k-ary relation from Γ. Then for every equivalence relation ρ on $\{1, \ldots, k\}$ either there is no $\sigma \in R$ that is coarser than ρ, or there exists an equivalence relation $\sigma \in R$ such that σ is coarser than ρ and σ is finer than any $\sigma' \in R$ coarser than ρ. Furthermore, σ can be computed in time $O(k^2|R|)$.*

Proof. First we compute the set R' of equivalence relations of R that are coarser than ρ. The set R' can be computed straightforwardly in time $O(k^2|R|)$ by checking each equivalence relation in R. If R' is empty we are done. Otherwise, because R is closed under intersections, we know that $\sigma = \cap_{\sigma' \in R'} \sigma'$ is in R. It is even in R', since if two equivalence relations are both coarser than another, then so is their intersection. We can find σ with the following procedure.

- We start with an arbitrary equivalence relation τ in R'.
- For each $\sigma' \in R'$, if σ' is finer than τ, then set τ to be σ'.

The procedure clearly runs in time $O(k^2|R|)$. $\square$

Theorem 6. *Let Γ be closed under a binary injective polymorphism, and let S be an instance of CSP(Γ) with n variables and q constraints. Let k be the maximal arity of the constraints, and let m be the maximal number of equivalence relations in the representations for the constraints. Then there is an algorithm that decides the satisfiability of S in time $O(qm(qmk^2 + n))$.*

Proof. We start by assigning each variable a unique value. Then we check whether each constraint is satisfied. If we find an unsatisfied k-ary constraint R, let

$x_1, \ldots, x_l$ be the variables of that constraint. Let ρ be the equivalence relation on the elements $\{1, \ldots, l\}$ that contains all pairs $\{i, j\}$ where x_i got the same value as x_j. Using the algorithm from Lemma 6 we either find that there is no $\sigma \in R$ coarser than ρ, in which case we answer that the problem does not have a solution. Otherwise we find the unique finest equivalence relation σ. In this case we reassign the values to the variables in the following way: If $\sigma(i, j)$, we assume without loss of generality that $i < j$, and change the value of all variables with the value of x_j to the value of x_i. Finally we restart the procedure with the new assignment for the variables. If all the constraints are satisfied we have computed a solution.

To show the correctness of this algorithm we prove by induction that each of the introduced equalities holds in every solution of the problem. In the beginning we introduced no equality (all the values were mutually different). We introduce an equality only if we find an unsatisfied constraint. In that case we have computed the set of equalities (an equivalence relation) that is contained in every other set of equalities acceptable for the constraint. Because the constraint must be satisfied in every solution we introduce only the equalities that hold in every solution.

Because the set of acceptable equivalence relations is made smaller each time the constraints are not yet satisfied, we have to recompute the assignment at most qm times. Finding the unsatisfied constraint can take $O(qmk^2)$ and changing the assignment can take $O(n)$. Putting the terms together yields the claimed bound on the time complexity. $\square$

Note that the asymptotic running time of the algorithm can be substantially improved by using better data structures. In the standard case that the signature of Γ is finite, the algorithm clearly establishes the tractability of CSP(Γ) for injective binary polymorphisms, since in this case k and m are bounded by constants that only depend on Γ.

If Γ has a countable signature, there are various possibilities to define tractability of CSP(Γ). We refer to the discussion in [6]. The definition of tractability chosen there is to require that for every reduct Γ' of Γ with a finite signature the problem CSP(Γ') is tractable. If Γ has an injective binary polymorphism, this requirement is clearly fulfilled, because we can again use the above algorithm with the same argument. If we allow that the instances contain arbitrary relations from the signature, we have to discuss how to represent the constraints in the instance.

For equality constraint languages, one natural candidate to represent the constraints in the instance is the representation that we already used in the formulation of the algorithm: a constraint is represented by a list of equivalence relations on its arguments. Now, the detailed complexity analysis given above shows that we even obtain tractability in the stronger sense where instances might contain arbitrary constraints in the above representation.

5 Conclusion and Remarks

Combining the results obtained in Sections 3 and 4 we proved Theorem 1, which can be reformulated as follows in the terminology of the algebraic approach to constraint satisfaction.

Theorem [Reformulation of Theorem 1]. An equality constraint language with template Γ is tractable if Γ has a constant unary or an injective binary polymorphism. Otherwise it is NP-complete.

We would like to conclude with a remark on the relationship of the presented results with questions from universal algebra. Clones that contain all the permutations are a recent research focus in universal algebra [9, 13, 14], and a full classification seams to be out of reach. However, the lattice of *locally closed* clones that contain the set of all permutations S_ω is considerably simpler. The lattice has a smallest element, the clone that is locally generated by S_ω. Above this clone, we have seen that it has exactly two minimal clones that correspond to the maximal tractable equality constraint languages. Is it possible to give a full description of the locally closed clones that contain all the permutations?

References

1. M. Bodirsky. The core of a countably categorical structure. In V. Diekert and B. Durand, editors, *Proceedings of the 22nd Annual Symposium on Theoretical Aspects of Computer Science (STACS'05), Stuttgart (Germany)*, LNCS 3404, pages 100–110, Springer-Verlag Berlin Heidelberg, 2005.
2. M. Bodirsky and J. Nešetřil. Constraint satisfaction with countable homogeneous templates. In *Proceedings of CSL'03*, pages 44–57, Vienna, 2003.
3. V. G. Bodnarčuk, L. A. Kalužnin, V. N. Kotov, and B. A. Romov. Galois theory for post algebras, part I and II. *Cybernetics*, 5:243–539, 1969.
4. A. Bulatov. Tractable conservative constraint satisfaction problems. In *Proceedings of LICS'03*, pages 321–330, 2003.
5. A. Bulatov, A. Krokhin, and P. Jeavons. The complexity of maximal constraint languages. In *Proceedings of STOC'01*, pages 667–674, 2001.
6. A. Bulatov, A. Krokhin, and P. G. Jeavons. Classifying the complexity of constraints using finite algebras. *SIAM Journal on Computing*, 34:720–742, 2005.
7. P. J. Cameron. *Oligomorphic Permutation Groups*. Cambridge University Press, 1990.
8. D. Geiger. Closed systems of functions and predicates. *Pacific Journal of Mathematics*, 27:95–100, 1968.
9. L. Heindorf. The maximal clones on countable sets that include all permutations. *Algebra univers.*, 48:209–222, 2002.
10. W. Hodges. *A shorter model theory*. Cambridge University Press, 1997.
11. P. Jeavons, D. Cohen, and M. Gyssens. Closure properties of constraints. *Journal of the ACM*, 44(4):527–548, 1997.
12. M. Krasner. Généralisation et analogues de la théorie de Galois. *Congrés de la Victoire de l'Ass. France avancement des sciences*, pages 54–58, 1945.

13. M. Pinsker. The number of unary clones containing the permutations on an infinite set. *Acta Sci. Math. (Szeged)*, 2005. To appear.
14. M. Pinsker. Precomplete clones on infinite sets which are closed under conjugation. *Monatsh. Math.*, 2005. To appear.
15. R. Pöschel and L. A. Kalužnin. *Funktionen- und Relationenalgebren*. Deutscher Verlag der Wissenschaften, 1979.