Monotone Monadic SNP and Constraint Satisfaction

Tomás Feder Moshe Y. Vardi
IBM Almaden Research Center
650 Harry Road
San Jose, California 95120-6099

Abstract

A constraint-satisfaction problem is given by a pair I (the *instance*) and T (the *template*) of finite relational structures over the same vocabulary. The problem is satisfied if there is a homomorphism from I to T. It is well-known that the constraint-satisfaction problem is NP-complete. In practice, however, one often encounters the situation where the template T is fixed and it is only the instance I that varies. We define *CSP* to be the class of constraint-satisfaction problems with respect to fixed templates. It is easy to see that CSP is contained in NP and that CSP contains both problems in P and NP-complete problems. We pose the question whether every problem in CSP is either in P or is NP-complete, and attempt to classify which problems in CSP are in P and which are NP-complete.

1 Introduction

A large class of problems in AI and other areas of computer science can be viewed as *constraint-satisfaction problems* [6, 18, 22]. This includes problems in machine vision, belief maintenance, scheduling, temporal reasoning, graph theory, and satisfiability. An instance of constraint satisfaction is given by a pair I, T of finite relational structures over the same vocabulary (the vocabulary is the list of relation names and their arities). The instance is satisfied if there is a homomorphism from I to T, that is, there exists a mapping h such that for every tuple $(x_1, \ldots, x_k) \in I$ we have $(h(x_1), \ldots, h(x_k)) \in T$. Intuitively, the elements of I should be thought of as variables and the elements of T should be thought of as possible values for the variables. The tuples in the relations of I and T should be viewed as constraints on the set of allowed assignments of values to variables. The set of allowed assignments is nonempty iff there exists a homomorphism from I to T.

It is well-known that the constraint-satisfaction problem is NP-complete. In practice, however, one often encounters the situation where the structure T (which we call the *template*) is fixed and it is only the structure I (which we call the *instance*) that varies.

For example, the template of the 3SAT problem has domain $\{0, 1\}$ and four ternary relations C_0, C_1, C_2, C_3 that hold for all triples except for $(0,0,0)$ in the case of C_0, except for $(1,0,0)$ in the case of C_1, except for $(1,1,0)$ in the case of C_2, and except for $(1,1,1)$ in the case of C_3. The tuples in the instance describe the clauses of the problem. For example, a constraint $C_2(x, y, z)$ imposes a condition on the three variables x, y, z that is equivalent to the clause $\overline{x} \vee \overline{y} \vee z$.

As a second example, the template of the 3-coloring problem is the graph K_3; i.e., it has domain $\{r, b, g\}$ and a single binary relation E that holds for all pairs (x, y) from the domain with $x \neq y$. The tuples in the instance describe the edges of

25th ACM STOC '93-5/93/CA,USA

the graph. Thus, the variables $x_1, x_2, \ldots, x_n$ can be viewed as vertices to be colored with r, b, g, and the constraints $E(x_i, x_j)$ can be viewed as describing the edges whose endpoints must be colored differently. If we replace the template K_3 by an arbitrary graph H, we get the so-called *H-coloring* problem [13].

As a third example, given an integer $q \geq 2$, the template of the linear equations modulo q problem has domain $\{0, 1, \ldots, q-1\}$, a monadic constraint Z that holds only for the element 0, and a ternary constraint C that holds for the triples (x, y, z) with $x+y+z = 1 \pmod{q}$. It is easy to show that any other linear constraint on variables modulo q can be expressed by introducing a few auxiliary variables and using only the Z and C constraints.

In this paper we consider constraint-satisfaction problems with respect to fixed templates. We define *CSP* to be the class of such problems. It is easy to see that CSP is contained in NP. We know that NP contains polynomially solvable problems and NP-complete problems. We also know that if P$\neq$NP, then there exist problems in NP that are neither in P nor NP-complete [19]. The existence of such "intermediate" problems is proved by a diagonalization argument. It seems, however, impossible to carry this argument in CSP. This motivates our main question:

Dichotomy Question: *Is every problem in CSP either in P or NP-complete?*

Our question is supported by two previous investigations of constraint-satisfaction problems that demonstrated dichotomies. Schaefer [26] showed that there are only three polynomially solvable constraint-satisfaction problems on the set $\{0, 1\}$, namely, Horn clauses, 2SAT, and linear equations modulo 2; all constraint-satisfaction problems on $\{0, 1\}$ that do not fit into one of these three categories are NP-complete. Hell and Nešetřil [13] showed that the H-coloring problem is in P if H is bipartite and NP-complete for H non-bipartite.

The issue that we address first is the robustness of the class CSP. We investigate the dichotomy question in the context of the complexity class *SNP*, which is a subclass of NP that is defined by means of a logical syntax [16, 24], and which, in particular, includes CSP. We show that SNP is too general a class to address the dichotomy question, because every problem in NP has an equivalent problem in SNP (in the sense that the two problems reduce to each other under polynomial time reductions). We then impose three syntactic restrictions on SNP, namely *monotonicity*, *monadicity*, and *no inequalities*, since CSP is contained in SNP with these restrictions imposed. It turns out that if only two of these three restrictions are imposed, then the resulting subclass of SNP is still general enough to contain an equivalent problem for every problem in NP.

When all three restrictions are imposed, we obtain the class *MMSNP*: *monotone monadic* SNP *without inequality*. This class is still more general than CSP. We prove, however, that every problem in MMSNP has an equivalent problem in CSP, this time under randomized polynomial time reductions (we believe that it may be possible to derandomize the reduction).

Thus, CSP is essentially the same as the seemingly more general class MMSNP. In the other direction, there are two special cases of CSP, namely, the *graph-retract* and the *digraph-homomorphism* problems, that turn out to be as hard as all of CSP, again under polynomial time reductions. The equivalence between CSP and classes both above and below it seems to indicate that CSP is a fairly robust class.

We then try to solve the dichotomy question by considering a more practical question:

Primary Classification Question: *Which problems in CSP are in P and which are NP-complete?*

In order to try to answer this question, we consider again Schaefer's results for constraint-satisfaction problems on the set $\{0, 1\}$ [26]. Schaefer showed that there are only three such polynomially solvable constraint-satisfaction problems. We introduce two subclasses of CSP, namely *bounded-width* CSP and *subgroup* CSP, respectively, as generalizations of Schaefer's three cases. Bounded-width problems are problems that can be solved by considering only bounded sets of variables, which we formalize in terms of the language Datalog [27]. Both Horn clauses and 2SAT fall into this subclass. Subgroup problems are group-theoretic problems where the constraints are expressed as subgroup constraints. Linear equations modulo 2 fall into this subclass. Not only are these subclasses solvable in polynomial time, but, at present, *all* known polynomially solvable constraint-satisfaction problems can be explained in terms of these conditions.

Assuming that these conditions are indeed the only possible causes for polynomial solvability for problems in CSP, this poses a new classification problem:

Secondary Classification Question: *Which problems in CSP are bounded-width problems and*

which are subgroup problems?

Our results provide some progress in understanding the bounded-width and subgroup subclasses. For example, for the bounded-width problems, our results provide a classification for the 1-width problems in CSP (these are the problems that can be solved by *monadic* Datalog programs). We also identify a property of problems, which we call *the ability to count*. We prove that this property implies that the problem cannot be solved by means of Datalog.

While all known polynomially solvable problems in CSP can be *explained* in terms of the bounded-width and group-theoretic subclasses, not all such problems *belong* to those classes. For example, we show that under some conditions non-subgroup problems can be *reduced* to the subgroup subclass. These conditions are stated in terms of the new notion of *nearsubgroup*, and delineating the boundary between polynomially solvable and NP-complete group-theoretical problems seems to require certain progress in finite-group theory.

2 Monotone Monadic SNP

The class SNP [16, 24] (see also [8]) consists of all problems expressible by an existential second-order sentence with a universal first-order part, namely, by a sentence of the form $(\exists S')(\forall \mathbf{x})\Phi(\mathbf{x}, S, S')$, where Φ is a first-order quantifier-free formula. Intuitively, the problem is to decide, for an input structure S, whether there exists a structure S' such that for all $\mathbf{x}$ it is true that $\Phi(\mathbf{x}, S, S')$ holds. We will refer to the relations of S as *input relations*, while the relations of S' will be referred to as *existential relations*. The 3SAT problem is an example of an SNP problem: The input structure S consists of four ternary relations C_0, C_1, C_2, C_3, where C_i corresponds to a clause on three variables with the first i of them negated. The existential structure S' is a single monadic relation T describing a truth assignment. The condition that must be satisfied states that for all x_1, x_2, x_3, if $C_0(x_1, x_2, x_3)$ then $T(x_1)$ or $T(x_2)$ or $T(x_3)$, and similarly for the remaining C_i by negating $T(x_j)$ if $j \leq i$.

It turns out that every problem in NP is equivalent to a problem in SNP under polynomial time reductions. In fact, we now show that this is the case even for restrictions of SNP. We start by assuming that the equality or inequality relations are not allowed. For *monotone* SNP, we require that all occurrences of an input relation C_i in Φ have the same polarity (the polarity of a relation is positive if it is governed by an even number of negations and it is negative otherwise); by convention, we assume that this polarity is negative, so that the C_i can be interpreted as constraints, in the sense that imposing C_i on more elements of the input structure can only make the instance "less satisfiable". Note that 3SAT as described above has this property. For *monadic* SNP, we require that the existential structure S' consist of monadic relations only. This is again the case for 3SAT described above. For *monotone monadic* SNP *with inequality*, we assume that the language contains also the equality relation, so both equalities and inequalities are allowed in Φ.

Theorem 1 *Every problem in NP has an equivalent (under polynomial time reductions) problem in monotone SNP, in monadic SNP, and in monotone monadic SNP with inequality. Therefore, if* P $\neq$ NP, *then there are problems in each of these subclasses of SNP that are neither in* P *nor* NP-*complete.*

We now consider the class *MMSNP*, which is *monotone monadic SNP, without inequality*. That is, in MMSNP we imposes all three restrictions simultaneously (instead of just two at a time as in the three subclasses of SNP considered above). It seems impossible to carry out Ladner's diagonalization argument in MMSNP. Thus, the dichotomy question from the introduction applies also to this class.

3 Constraint Satisfaction

Let S and T be two finite relational structures over the same vocabulary. A *homomorphism* from S to T is a mapping from the elements of S to elements of T such that all elements related by some relation C_i in S map to elements related by C_i in T. If T is a restriction of S to a subset of the elements, and the homomorphism from S to T is just the identity mapping when restricted to T, then the homomorphism is called a *retraction*, and T is called a *retract* of S. If no proper restriction T of S is a retract of S, then S is a *core*, otherwise its core is a retract T that is a core (unique up to isomorphism).

A *constraint-satisfaction problem* will be here a problem of the following form. Fix a finite relational structure T over some vocabulary; T is called the *template*. An *instance* is a finite relational structure S over the same vocabulary.

The instance is satisfied if there is a homomorphism from S to T. Such a homomorphism is called a *solution*. We define the *CSP* to be the class of constraint-satisfaction problems. (We remark that it is possible to define constraint satisfaction with respect to infinite templates. For example, digraph acyclicity can be viewed as the question of whether a given digraph can be homomorphically mapped to the transitive closure of an infinite directed path. We will not consider infinite templates in this paper.)

It is easy to see that CSP is contained in MMSNP. Let T be a template. Then there is monadic monotone existential second-order sentence ϕ_T (without inequality) that expresses the constraint-satisfaction problem defined by T. For each element a in the domain of the template T, we introduce an existentially quantified monadic relation T_a; intuitively, $T_a(x)$ indicate that a variable x has been assigned value a by the homomorphism. The sentence ϕ_T says that the sets T_a are disjoint and that the tuples of S satisfy the constraints given by T.

It can be shown that CSP is strictly contained in MSNP. Nevertheless, as the following theorem shows, in terms of the complexity of its problems, CSP is just has hard as MMSNP.

Theorem 2 *Every problem in MMSNP is polynomially equivalent to a problem in CSP. The equivalence is by a randomized Turing reduction from CSP to MMSNP and by a deterministic Karp reduction from MMSNP to CSP.*

The construction in the preceding reduction from monotone monadic SNP to constraint-satisfaction problems can also be used to show the following result, which will be proven useful later.

Theorem 3 *Containment is decidable for problems in MMSNP.*

The *graph-retract problem* is an example of a constraint-satisfaction problem. Fix a graph H, and for an input graph G containing H as a subgraph, ask whether H is a retract of G. (Note that when G and H are disjoint, we get the *graph homomorphism* or H-coloring problem mentioned in the introduction [13].)

Theorem 4 *Every constraint-satisfaction problem is polynomially equivalent to a graph-retract problem.*

The *digraph-homomorphism problem* is another example of a constraint-satisfaction problem.

Theorem 5 *Every constraint-satisfaction problem is polynomially equivalent to a digraph-homomorphism problem.*

Therefore, the dichotomy question for MMSNP is equivalent to the dichotomy question for CSP, which in turn is equivalent to the dichotomy questions for graph-retract and digraph-homomorpshim problems.

4 Special Classes

Schaefer [26] showed that there are only three polynomially solvable constraint-satisfaction problems on the set $\{0,1\}$, namely Horn clauses, 2SAT, and linear equations modulo 2; all constraint-satisfaction problems on $\{0,1\}$ that do not fit into one of these three categories are NP-complete. For general constraint-satisfaction problems, we introduce three classes, namely *bounded width*, *bounded strict width*, and *subgroup*, respectively, as generalizations of these three cases. At present, all known polynomially solvable constraint satisfaction problems are simple combinations of the bounded-width case and the subgroup case. (Note: a similar situation of only three polynomially solvable cases was observed for Boolean network stability problems by Mayr and Subramanian [23], Feder [9]; the three cases there are monotone networks, linear networks, and nonexpansive networks, in close correspondence with Horn clauses, linear equations modulo 2, and 2SAT respectively; it is the generalization of the nonexpansive case to metric networks that leads to characterizations along the lines of the bounded strict width case described below; in fact, the restrictiveness of problems that are simultaneously group-theoretic and of bounded strict width, mentioned below, seems to be related to the simplicity of isomorphisms on the space of solutions for nonexpansive metric network stability.)

4.1 Bounded-Width Problems

A constraint-satisfaction problem is said to have *bounded width* if its complement (i.e., the question of non-existence of a solution) can be expressed in Datalog. More precisely, it is said to have *width* (l,k) if the corresponding Datalog program has rules with at most l variables in the head and at most k variables per rule, and is said to have *width* l if it is has width (l,k) for some k. (For a related notion of width, see Afrati and Cosmadakis [2].)

Datalog is the language of logic programs without function symbols [27]. The following Datalog

program checks that an input graph is not 2-colorable:

$$\begin{array}{lcl} \mathit{evenpath}(X,Y) & :- & \mathit{edge}(X,Z), \mathit{edge}(Z,Y) \\ \mathit{evenpath}(X,Y) & :- & \mathit{evenpath}(X,Z), \mathit{evenpath}(Z,Y) \\ \mathit{not2colorable} & :- & \mathit{evenpath}(X,X). \end{array}$$

In this example, *edge* is an input binary relation, *evenpath* is a binary relation computed by the program, and *not2colorable* is a 0-ary relation computed by the program. The first rule says that every pairs of adjacent edges forms an even path, the second rule tells that even paths can be joined together to form an even path, and the third rule says that the input graph is not 2-colorable if the graph contains an even cycle. In Datalog programs for constraint-satisfaction instances we assume that there is a distinguished predicate p of arity zero (*not2colorable* in the above example) that must be derived when no solution exists for the instance. We say that such a program *solves* the problem.

The example above shows that 2-colorability has width $(2,3)$, since it can be solved by a Datalog program with at most 2 variables in rule heads and at most 3 variables per rule. Also, 3SAT-Horn can be shown to have width 1. It is not hard to show that bounded-width CSP is contained in monotone SNP without inequality. Furthermore, constraint-satisfaction problems of width 1 are in MMSNP.

It is easy to see that all bounded-width constraint-satisfaction problems are in P, since the rules can derive only a polynomial number of facts. Thus, we'd like to know:

Which problems in CSP have bounded width?

Even if we know that a constraint-satisfaction problem has width (l,k), there could be many Datalog programs that expressed the complement of the problem. Thus, it seems that to answer the question above we need to consider all possible (l,k)-programs. Surprisingly, it suffices to focus on very specific Datalog programs.

Theorem 6 *For every constraint satisfaction problem P there is a canonical Datalog (l,k)-program with the following property: if any Datalog (l,k)-program solves P, then the canonical one does.*

Intuitively, the canonical program width (l,k) infers all possible constraints on l variables at a time by considering k variables at a time.

Using the above theorem and Theorem 3, we can prove:

Theorem 7 *The questions of whether a constraint-satisfaction problem has width $(1,k)$ or whether it has width 1 are decidable.*

In general, the question of whether a constraint-satisfaction problem has bounded width (or width l, width (l,k), beyond the case $l=1$) is not known to be decidable.

Which problems in CSP do not have bounded width? Say that a constraint-satisfaction problem has *the ability to count* if the following holds. The structure T contains at least the values $0,1$, as well as a ternary relation C that includes at least the triples $(1,0,0)$, $(0,1,0)$, and $(0,0,1)$, as well as a monadic relation Z that accepts at least 0. We also assume that if an instance consists of only the constraints C and Z with all constraints partitioned into two sets A and B such that A contains one more C constraint than B, and furthermore each variable appears in two constraints, one from A and one from B, then the instance has no solution. Intuitively, we can think of $C(x,y,z)$ as $x+y+z=1$, and of $Z(x)$ as $x=0$, with an obvious contradiction for instances of the special form, since adding the constraints from A and subtracting those from B yields $0=1$. Some problems of this form include linear equations over an abelian group (finite or infinite), where 1 is any element other than the zero of the group. Schaefer [26] In particular, this includes the problem for linear equations modulo 2. Another example of such problem is linear programs over the nonnegative reals. For this last case, inexpressibility in Datalog was shown by Afrati et al. [3] using Razborov's monotone circuit lower bound for matching [25].

Theorem 8 *If a constraint-satisfaction problem has the ability to count, then it does not have bounded width.*

Thus, the polynomial solvability of linear equations modulo 2 cannot be explained in terms of Datalog.

It seems possible that the ability to count is the converse of having bounded width. The intuition for this is that the non-existence of solutions can be attributed to the presence of the same variable in different constraints, something that cannot be remembered by Datalog if these occurrences in two different places are not ordered, and it seems that to keep track of equality of variables in different order one needs the ability to count.

4.2 Bounded Strict Width Problems

The canonical algorithm for problems of width (l,k) involved inferring all possible constraints on

l variables at a time by considering k variables at a time. We may in addition require that if this inference process does not reach a contradiction (the empty set), then it should be possible to obtain a solution by greedily assigning values to the variables one at a time while satisfying the inferred l-constraints. We say that a constraint-satisfaction problem that can be solved in this way has *strict width* (l, k), and say that it has *strict width* l if it has strict width (l, k) for some k. It turns out that strict width l is equivalent to strict width (l, k), for all $k > l$, so we can assume $k = l + 1$.

This intuition behind strict width l can also be captured in two other ways. First, we can require that if we have an instance and after assigning specific values to some of the variables we obtain an instance with no solution, then some l out of the specific value assignments chosen are sufficient to give an instance with no solution. We refer to this property as the l-Helly property. Second, we could require that there exists a function g that maps $l + 1$ elements from the domain of the structure T to another element, with the property that if all but at most one of the $l + 1$ arguments are equal to some value b then the value of g is also b, and, furthermore, for all constraints C_i in T, if we have $l + 1$ tuples satisfying C_i, then the tuple obtained by applying g component-wise also satisfies C_i. We call this property the l-mapping property.

Theorem 9 *Strict width l, the l-Helly property, and the l-mapping property are equivalent. These properties are polynomially decidable (for a fixed l).*

2-colorability is an example of a constraint-satisfaction problem with strict width 2. If a graph is bipartite, but after having two-colored some of the vertices there is no two-coloring consistent with this partial coloring, then either two vertices on different sides of the bipartite graph were given the same color, or two vertices in the same side were given different colors. Also, 2SAT has strict width 2, and so does integer programming with two variables per inequality with variables ranging over a fixed range.

The third equivalent definition is based on the following observation. For 2SAT instances, if we have three solutions x, y, z, to an instance, and we define an assignment $t = \text{median}(x, y, z)$ by $t_i = \text{majority}(x_i, y_i, z_i)$, then t is a solution as well. In fact, any set of assignments closed under median can be characterized by a 2SAT instance.

Problems with bounded strict width are a special case of bounded-width CSP. For such problems we have a more efficient algorithm.

Theorem 10 *A simplified version of the canonical algorithm runs in parallel $O^*(n)$ time. In the case $l = 2$, this algorithm can be implemented in parallel $O^*(\sqrt{n})$ time, because variables can be eliminated in parallel.*

The encoding of constraint-satisfaction problems as retract problems or digraph homomorphism problems preserves the property of having width l (perhaps not the property of having width (l, k)). Note that a problem of width 1 will necessarily have width $(1, 2)$ when encoded, since k need not be larger than the arity. Strict width is preserved by the encoding as a digraph homomorphism problem, but a problem may lose the bounded strict width property when encoded as a retract problem, unless we allow a vertex in H whose neighborhood is a proper subset of the neighborhood of another vertex to be used as a value in the input graph but not as a value for the remaining vertices in the solution.

We can construct problems having strict width l requiring l to be exponential in the size of the domain of the template. For example, consider the structure T whose domain is pairs (i, j), $0 \leq i \leq r$, $1 \leq j \leq 3$. Monadic constraints accept the pairs with i fixed. A binary relation does not relate (i, j) with (i', j') only if $i' = i+1$, $j = 1$, $j' \neq 1$. Consider the case where all arguments to g have the same i. Suppose that g cannot output (i, j) if $j \neq 1$ appears in at most k argument positions. Then g cannot output $(i + 1, j')$ if $j' \neq 1$ appears in at most $2k$ argument positions. So $l \geq 2^{r-1} + 1$ by induction, and one can design g for such l. We do not know whether l may need in some cases to grow even faster in the size of the domain. These problems, however, turn out to have width 1 or 2, for instance, width 1 in the example. In fact, it may be that all bounded width problems are just combinations of problems of width 1 and problems of strict width 2, via products and simple encodings. We know of no bounded width problem that does not have width 2, or even width $(2, 3)$ in the case of digraph homomorphism.

4.3 Subgroup Problems

A constraint-satisfaction problem is *group-theoretic* when the elements of the template T are elements in a finite group H. When each k-ary relation in T is a coset of a subgroup of the direct product H^k, we call it a *subgroup* constraint, the problem can be shown to be in P (Babai [4], Furst et al. [11]; see also Theorem II.12 in Hoffmann [15]). We call such

problems *subgroup* problems. In other words, in a subgroup problem the variables $x_1, \ldots, x_n$ range over the elements a finite group H, and on a tuple of variables of arity k, we may impose the constraint, $(x_{i_1}, \ldots, x_{i_k}) \in S$, where S is a chosen coset of some subgroup of the direct product H^k.

The main observation is that given a group G with known generators and a chain of subgroups $G = G_1 > G_2 > \cdots > G_r = \{1\}$, one can obtain distinct representatives from each coset of each G_i in G_{i-1} as follows. Select two elements x, x' among the generators of G_1 that belong to the same coset of G_2, say $x' = xy$ with $y \in G_2$, then discard x' and add y to the list of generators. Iterate until there is only one generator in each coset of each G_i in G_{i-1}, and carry out the process for products xy of two current generators as well. (The fact that only products of pairs are needed to obtain representatives for all cosets of each G_i in G_{i-1} requires proof, see Theorem II.8 in [15].) In our application, $G = H^n$ has known generators, the G_i are obtained by adding each of the subgroup constraints, until some subgroup $J = G_j$ is obtained, then the remaining G_i are obtained by fixing the n components to 1 successively, until $G_r = G_{j+n} = \{1\}$ is obtained. To solve the constraint-satisfaction problem, observe that the first constraint selects a coset of G_2 in G_1, so we may select a representative y from this coset and look for a solution of the form xy with x in G_2, after determining to which coset x must belong for each of the remaining constraints. This gives a polynomial time algorithm because n, r, $|H|$, and $|G_i|/|G_{i-1}|$ are polynomially bounded.

This class contains as a special case linear equations for abelian groups. The main example of the class is labelled graph isomorphism, where two graphs have been colored with each color occurring a bounded number of times, and we look for a color-preserving isomorphism. There H is the group of permutations on vertices that have all the same color, and the constraints require vertices in one graph to map to vertices in the other (this is a subset constraint on H) and that edges be preserved (this is a subgroup constraint on H or H^2, depending on whether the two endpoints of the edge have the same or different colors).

The main question that we want to address now is what happens when we go slightly beyond subgroup problems, by adding non-subgroup constraints. We first consider abelian groups.

Theorem 11 *Group-theoretic constraint-satisfaction problems over finite abelian groups are NP-complete when the template includes a non-subgroup constraint.*

We now consider the non-abelian case. Assume without loss of generality that K contains the identity 1. We say that K is a *nearsubgroup* of G if for all b, x, y, if $b, bx, by \in K$, then there is some z in the commutator group of the subgroup generated by $\{x, y\}$, such that $bxyz \in K$.

Theorem 12 *Group-theoretic constraint-satisfaction problems are NP-complete when the template includes a non-nearsubgroup constraint.*

The case analysis in the study of the abelian case reduced the study of subsets that are not subgroups to two cases, one of them one-dimensional, involving a subset containing $0, 1$ but not 2, the other one two-dimensional, involving a subset containing $(0,0), (0,1), (1,0)$ but not $(1,1)$. From a similar discussion, the property that K is a nearsubgroup reduces to (1) the cycles condition, if x, y satisfy K then so does $xy^{-1}x$, or equivalently xyx (since we assume $1 \in K$), and (2) if H is a subgroup of G with a normal subgroup N and $H/N \approx Z_2^2$, then it is not the case for some b that bK intersects exactly three of the four cosets of N in H. We ask whether the group-theoretic problem is in P when one of the constraints is a nearsubrgoup constraint.

In fact, we can replace (2) by the condition that if $b, bx, by \in K$ then there is some z in the commutator group of the subgroup generated by $\{x, y\}$, such that $bxyz \in K$, but require this only when the orders of x, y are powers of 2. The reason is that we can set $x' = x^r$, $y' = y^s$, with r and s odd, and $bx', by' \in K$ by the cycles condition, but $bx'y'z' \notin K$, since r, s both odd was not in K in the abelian case once the cycles condition holds. We can also require that the order of xyz be a power of 2, because one can choose k odd such that t^k has order a power of 2 for all t and $t^k = t$ when the order of t is a power of 2; then $(xyz)^k = xyz'$ has order a power of 2. We do not know whether we may just need to consider the case where x, y generate a 2-group, in which case bxy itself must be in K (see below). A weaker property that is also open is whether the intersection of two nearsubgroups is a nearsubgroup.

Theorem 13 *For 2-groups (groups of order a power of 2), if K is a nearsubgroup, then K is a subgroup.*

Theorem 14 *If a template over a finite group of odd order contains subgroup constraints and*

one nearsubgroup constraint, then the constraint-satisfaction problem is in P.

In fact, the argument does not depend on the group having odd order, only on the constant a having odd order (this constant is the only constraint not containing 1 in the proof). Then $a^{2^i} = a$ for some $i \geq 1$, so every solution gives a new solution that replaces each value z by z^{2^i}. There is therefore a solution and a $j \geq 1$ such that $z^{2^j} = z$ for all values z. Then if $z \in K$, then $z^{\frac{1}{2}} \in K$, so a solution to the original problem gives a solution for the new problem as well; the converse goes through as before. In fact, a constant a can be replaced by a variable x that ranges over the cyclic group generated by a, which can in turn be decomposed as the product of an odd order cyclic group and a cyclic group whose order is a power of 2, with the constant a replaced by two constants in the two cyclic groups. The constant of odd order is handled as before, while when the order of a is a power of 2, we may by the same argument, with $a^{k^i} = a$ and k odd, consider just solutions with the order of all z a power of 2.

We use this to show that the case of a nearsubgroup is in P for a dihedral group, e.g., the constraint $x^2 = 1$ on S_3. The dihedral group D_{2n} of order $2n$ is generated by c, d with $c^2 = d^n = 1$ and $cd = d^{-1}c$. By the cycles condition, K is a subgroup unless it contains some cd^i, say c. Then K contains elements d^{ki} and cd^{rj}, for some $k, r|n$. Also, because K contains c, cd^r, it contains some commutated product $d^{r(2i+1)}$, and since K contains c, d^k, it contains some $cd^{k(2j+1)}$, so the same power of 2 divides k and r. The elements in K of order a power of 2 are then the elements of order a power of 2 in the subgroup generated by c and d^r, so K can be viewed as a subgroup constraint. (Thus $x^{2k} = 1$ on D_{2n} is in P if the highest power of 2 dividing n divides $2k$, and NP-complete otherwise.)

The remaining case is summarized in the following theorem.

Theorem 15 *Let D contain the elements of G, whose order is a power of 2, an instance consists of a constant a from D, subgroup constraints, and a nearsubgroup K. Here one may restrict variables to range over D. Letting S be the intersection of K with D, and letting S' be the intersection of the subgroup generated by S with D, it can be that $S = S'$. A positive answer here makes the group-theoretic problem be in P for nearsubgroups. This is the case for groups of order a power of 2, groups with Sylow 2-subgroups Z_2^n, and dihedral groups.*

The case of Sylow 2-subgroups Z_2^n was shown by elementary arguments in some cases ($n = 0, 1, 2$), but follows in general directly from the two theorems below due to Aschbacher [1]. (The first one presented here with slightly greater generality; Aschbacher originally showed it with the stronger assumption that G contains a strongly embedded subgroup, and this is for instance the case for the Bender groups, i.e., the simple groups of Lie type of even characteristic and Lie rank 1.) Let I denote the involutions plus 1, i.e., all z satisfying $z^2 = 1$.

Theorem 16 *The set I is a nearsubgroup of a group G if and only if G contains no D_8 subgroup.*

Theorem 17 *Let I denote the involutions plus 1 in a nearsubgroup K of G, and suppose that I generates G. Then there is a normal subgroup H such that $xH \subseteq K$ for each $x \in K$, and the induced subset K^* of $G^* = G/H$ is a nearsubgroup of G^* that contains precisely all the involutions plus 1. (Here H is the subgroup generated by the squares of elements in K.)*

The case of Sylow 2-subgroups Z_2^n follows from this last theorem. The intersection $I = K \cap D$ contains only involutions and 1, and generates some subgroup G. Let $K' = K \cap G$. If G contains some involution not in K', then $G^* = G/H$ contains some involution not in K'^*, so K' is not a nearsubgroup, a contradiction.

Theorem 16, on the other hand, provides examples of nearsubgroups that violate the above condition for a polynomial time algorithm, namely I in any group with no D_8 subgroup, but where I generates some element of order 4. In the case of simple groups, it is sufficient that the group contain some element of order 4, since the involutions form a conjugacy class and hence generate the group. The counterexamples are rank 1 simple Lie groups of even characteristic with at least one element of order 4.

5 Further Directions

As an illustration of the fact that only combinations of width 1, strict width 2 and group-theoretic are known, we consider a conjecture of Bang-Jensen and Hell [5]. This conjecture states that if a digraph is a core and has no sources or sinks, then the associated digraph homomorphism problem is in P if the digraph is a cycle, and NP-complete otherwise. If it holds, it would extend the result of Hell and Nešetřil for graph homomorphism, since

a graph is a digraph with edges oriented in both directions. Consider the test for width 1. This test involves all nonempty subsets A of the domain that can be inferred by the canonical algorithm for width 1. In the simplest scenario, the only such set may be the entire domain A; for the case of a single binary relation (a digraph), this happens when every element is related to some other element in both directions, i.e., when the digraph has no sources or sinks. If the digraph has no self-loops (otherwise the associated core is a self-loop), we can infer that the problem does not have width 1, since the instance obtained by imposing the relation on the single element x_A has no solution. We may then look for problems of strict width 2 and for group-theoretic problems. The conjectured polynomial cases are equations of the form $y = x + 1$ modulo q, hence simultaneously group-theoretic and of strict width 2. This suggests considering hypergraph homomorphism, where the edges are unordered multisets of size r. Here again, every element is related to some $r - 1$ elements, regardless of the position of that element since edges are unordered, so beyond the trivial case where the hypergraph has a self-loop (an edge with r copies of the same element), the problem does not have width 1. It seems then that this problem is not as general as digraph homomorphism, because digraph homomorphism has problems of width 1, e.g., the encoding of Horn clauses; and furthermore, encodings tend to preserve width. It may be that all hypergraph homomorphism problems are either NP-complete or group-theoretic. An example of a group-theoretic problem here is defined by $x_1 + x_2 + \ldots + x_r = 1$ modulo r. A case that can be handled directly has all r elements distinct for all edges; then we may consider instances that are graphs, where each edge is augmented with $r - 2$ extra elements, giving NP-completeness by the result of Hell and Nešetřil.

Say that a core T *can simulate* a core T' if for every relation C_i in T' there is an instance S_i that defines on some variables in S_i a relation C_i' over the domain of T whose core is precisely C_i. (Here we can bound $|S_i| \leq |T|^{|C_i|}$.)

Conjecture 1 *A constraint-satisfaction problem is not in Datalog if and only if the associated core T can simulate a core T' consisting of two relations C, Z that give the ability to count. This is equivalent to simulating either Z_p or one-in-three SAT.*

Conjecture 2 *A constraint-satisfaction problem is NP-complete if and only if the associated core T can simulate a core T' consisting of the single relation C defining one-in-three SAT.*

As an example, the proofs of NP-completeness for non-bipartite H-coloring simulate 3-coloring [13], which in turn can simulate one-in-three SAT.

According to the first conjecture, once a problem cannot be solved with Datalog, it has the ability to count, and hence some group-theoretic element. The intuition here is that the non-existence of solutions can be attributed to the presence of the same variable in different constraints, something that cannot be remembered by Datalog if these occurrences in two different places are not ordered, and it seems that to keep track of equality of variables in different order one needs the ability to count using Z_p.

Conjecture 3 *If a constraint-satisfaction problem is not NP-complete, and can simulate a group-theoretic problem, then it is possible to view the problem as a combination of a Datalog problem and a group-theoretic problem with nearsubgroup constraints, by means of products and encodings (e.g., the product of 2SAT and linear equations modulo 2, or the encoding of linear equations modulo 2 as a digraph homomorphism problem).*

Conjecture 4 *Any problem whose constraints are cosets of subgroups or nearsubgroups is polynomially solvable. In particular, when the involutions plus 1 form a nearsubgroup, they define a polynomially solvable group-theoretic problem.*

Conjecture 5 *The intersection of two nearsubgroups is a nearsubgroup.*

A stronger conjecture would state that a set containing 1 is a nearsubgroup if it satisfies the cycles condition and its intersection with a coset of a 2-group is a coset of a subgroup (a valid definition for current results).

The relevance of this last conjecture lies in the fact that if all constraints are subgroups or nearsubgroups, new constraints defined from them will be nearsubgroups, therefore one-in-three SAT cannot be defined and by Conjecture 2 the problem is unlikely to be NP-complete.

Conjecture 6 *All constraint-satisfaction problems are either polynomially solvable or NP-complete. The polynomially solvable constraint-satisfaction problems have complexity polynomial in the size of the instance and of the template as well, for constraints of fixed total arity.*

This conjecture would follow from the previous ones. All conjectures are direct statements that may be provable directly, except for Conjecture 2, which would require means of proving that a problem is not NP-complete, and Conjecture 3, which contains an ill-defined notion of encoding.

The intuition for a classification seems to extend outside monotone monadic SNP. We may consider infinite but well-structured domains, e.g., constraints described by nice functions over the reals. Here it also seems that for a problem not to be of bounded width, it may need the ability to count, but not necessarily modulo q, as above for linear programs related to matching; it may then be that adding extra constraints to a problem that can count makes it NP-complete. The class SNP itself seems to have the roots of a dichotomy. For example, monotone binary SNP contains the acyclicity problem. Versions of graph isomorphism that can be expressed in SNP (not monotone monadic) specify a bounded number of candidate images for each vertex ahead of time (NP-complete [21]) or require the graphs to have bounded degree (polynomial [4, 11]). The direct encoding of graph isomorphism in SNP seems unnatural, in that it requires an auxiliary path going through all the vertices; we would like a restriction on SNP that is weaker than monotone monadic, yet does not allow imposing an auxiliary path structure on a problem. It is worth noting here the distinction with the syntactic definition of NP [8], where NP problems on structures can be expressed directly, with no encoding. Locality seems important, in that it only allows a few possible values for each element. For example, in graph isomorphism, once a vertex has been mapped in the solution, its neighbors can only map to neighbors, and the number of possibilities is small if the graph has bounded degree. For network stability problems, nonexpansiveness requires close vertices to map to close vertices in the instance, and the L_1 metric makes neighborhoods small.

Say that a problem A in NP is *minimal* if there is no problem B equivalent to A under polynomial time reductions (i.e., polynomial time reductions in both directions) such that the reduction from A to B reduces all instances from size n to size $n - n^\epsilon$ for some $\epsilon > 0$. Notice that the known encodings of graph isomorphism in SNP are not minimal. Similarly, for stability in nonexpansive networks with integer values, the input seems to require extra space to allocate the integer values in the solution, so the natural encoding is not minimal.

Conjecture 7 *All minimal problems in SNP with inequality are NP-complete.*

Acknowledgements

We benefitted greatly from early discussions with Yatin Saraiya and with Peter Winkler. Christos Papadimitriou suggested the connection between constraint satisfaction and two problems with a fixed structure that were previously studied, graph homomorphism and the boolean domain case. Milena Mihail suggested that quasirandom graphs may derandomize the representation of monotone monadic SNP in CSP. We also had very valuable conversations with Miki Ajtai, Michael Aschbacher, Laszlo Babai, Ron Fagin, Jim Hafner, Pavol Hell, Rajeev Motwani, and Moni Naor.

References

[1] M. Aschbacher, personal communication.

[2] F. Afrati and S. S. Cosmadakis, "Expressiveness of restricted recursive queries," Proc. 21st ACM Symp. on Theory of Computing (1989), 113–126.

[3] F. Afrati, S. S. Cosmadakis, and M. Yannakakis, "On Datalog vs. polynomial time," Proc. 10th ACM SIGACT-SIGMOD-SIGART Symp. on Principles of Database Systems (1991), 13–25.

[4] L. Babai, "Monte Carlo algorithms in graph isomorphism testing," (1979).

[5] J. Bang-Jensen and P. Hell, "The effect of two cycles on the complexity of colourings by directed graphs," Discrete Applied Math 26 (1990), 1–23.

[6] R. Dechter, "Constraint networks," In Encyclopedia of Artificial Intelligence, 1992, 276–285.

[7] P. Erdős, "Graph theory and probability," Canadian J. of Math. 11 (1959), 34–38.

[8] R. Fagin, "Generalized first-order spectra, and polynomial-time recognizable sets," in R. Karp (ed.), Complexity of Computations, AMS, 1974.

[9] T. Feder, "Stable Networks and Product Graphs," doctoral dissertation, Stanford University (1991).

[10] T. Feder, "Removing inequalities and negation for homomorphism-closed problems," in preparation.

[11] M. Furst, J. E. Hopcroft, and E. Luks, "Polynomial-time algorithms for permutation groups," Proc. 21st IEEE Symp. on Found. of Comp. Sci. (1980), 36–41.

[12] D. M. Goldschmidt, "2-fusion in finite groups," Annals of Math. 99 (1974), 70–117.

[13] P. Hell and J. Nešetřil, "On the complexity of H-coloring," J. Comb. Theory, Series B 48 (1990), 92–110.

[14] G. G. Hillebrand, P. C. Kanellakis, H. G. Mairson, and M. Y. Vardi, "Tools for Datalog boundedness," Proc. 10th ACM SIGACT-SIGMOD-SIGART Symp. on Principles of Database Systems (1991), 1–12.

[15] C. M. Hoffmann, "Group-Theoretic Algorithms and Graph Isomorphism," Lecture Notes in Comp. Sci. 136 (1982), Springer-Verlag.

[16] P. G. Kolaitis and M. Y. Vardi, "The decision problem for the probabilities of higher-order properties," Proc. 19th ACM Symp. on Theory of Computing (1987), 425–435.

[17] P. G. Kolaitis and M. Y. Vardi, "On the expressive power of Datalog: tools and a case study," Proc. 9th ACM SIGACT-SIGMOD-SIGART Symp. on Principles of Database Systems (1990), 61–71.

[18] V. Kumar, "Algorithms for constraint-satisfaction problems," AI Magazine 13 (1992), 32–44.

[19] R. E. Ladner, "On the structure of polynomial time reducibility," J. Assoc. Comput. Mach. 22 (1975), 155–171.

[20] V. S. Lakshmanan and A. O. Mendelzon, "Inductive pebble games and the expressive power of Datalog," Proc. 8th ACM SIGACT-SIGMOD-SIGART Symp. on Principles of Database Systems (1989), 301–310.

[21] A. Lubiw, "Some NP-complete problems similar to graph isomorphism," SIAM J. Comput. 10 (1981), 11–21.

[22] P. Meseguer, "Constraint satisfaction problem: an overview," AICOM 2 (1989), 3–16.

[23] E. Mayr and A. Subramanian, "The complexity of circuit value and network stability," J. Comput. Syst. Sci. 44 (1992), 302–323.

[24] C. H. Papadimitriou and M. Yannakakis, "Optimization, approximation, and complexity classes," J. Comput. Syst. Sci. 43 (1991), 425–440.

[25] A. A. Razborov, "Lower bounds on monotone complexity of the logical permanent," Math. Notes of the Academy of Sciences of the USSR 37 (1985), 485–493.

[26] T. J. Schaefer, "The complexity of satisfiability problems," Proc. 10th ACM Symp. on Theory of Computing (1978), 216–226.

[27] Ullman, J.D.: *Principles of Database and Knowledge-Base Systems*, Vol I. Computer Science Press, 1989.