Constraint Propagation as a Proof System

Albert Atserias[⋆1], Phokion G. Kolaitis[⋆⋆2], and Moshe Y. Vardi[⋆⋆⋆3]

[1] Universitat Politècnica de Catalunya, Barcelona, Spain
[2] University of California, Santa Cruz, USA
[3] Rice University, Houston, USA

Abstract. Refutation proofs can be viewed as a special case of constraint propagation, which is a fundamental technique in solving constraint-satisfaction problems. The generalization lifts, in a uniform way, the concept of refutation from Boolean satisfiability problems to general constraint-satisfaction problems. On the one hand, this enables us to study and characterize basic concepts, such as refutation width, using tools from finite-model theory. On the other hand, this enables us to introduce new proof systems, based on representation classes, that have not been considered up to this point. We consider ordered binary decision diagrams (OBDDs) as a case study of a representation class for refutations, and compare their strength to well-known proof systems, such as resolution, the Gaussian calculus, cutting planes, and Frege systems of bounded alternation-depth. In particular, we show that refutations by ODBBs polynomially simulate resolution and can be exponentially stronger.

1 Introduction

It is well known that the satisfiability problem for Boolean formulas in conjunctive normal form (CNF) can be viewed as a *constraint-satisfaction problem* (CSP). The input to a CSP consists of a set of variables, a set of possible values for the variables, and a set of constaints on the variables. The question is to determine whether there is an assignment of values to the variables that satisfies the given constraints. The study of CSP occupies a prominent place in artificial intelligence and computer science, because many algorithmic problems from a wide spectrum of areas can be modeled as such [Dec03]. These areas include temporal reasoning, belief maintenance, machine vision, scheduling, graph theory, and, of course, propositional logic. Since constraint-satisfaction problems constitute a natural generalization of Boolean satisfiability problems, it is natural to ask for proof systems that generalize the systems for propositional logic to CSP. Such systems would be used to refute the satisfiability of an instance of a constraint-satisfaction problem, much in the same way that resolution is used to refute the satisfiability of a CNF-formula.

⋆ Supported in part by CICYT TIC2001-1577-C03-02 and the Future and Emerging Technologies programme of the EU under contract number IST-99-14186 (ALCOM-FT).
⋆⋆ Supported in part by NSF grant IIS-9907419.
⋆⋆⋆ Supported in part by NSF grants CCR-9988322, CCR-0124077, CCR-0311326, IIS-9908435, IIS-9978135, EIA-0086264, and ANI-0216467, and by BSF grant 9800096.

One of the goals of this paper is to introduce a natural and canonical way of defining a proof system for every constraint-satisfaction problem. In order to achieve this, first we need a unifying framework for representing such problems. This was achieved by Feder and Vardi [FV98], who recognized that essentially all examples of CSPs in the literature can be recast as the following fundamental algebraic problem, called the HOMOMORPHISM PROBLEM: given two finite relational structures $\mathbf{A}$ and $\mathbf{B}$, is there a homomorphism $h : \mathbf{A} \to \mathbf{B}$? Intuitively, the structure $\mathbf{A}$ represents the variables and the tuples of variables that participate in constraints, the structure $\mathbf{B}$ represents the domain of values, and the tuples of values that these constrained tuples of variables are allowed to take, and the homomorphisms from $\mathbf{A}$ to $\mathbf{B}$ are precisely the assignments of values to variables that satisfy the constraints. For instance, the 3-COLORABILITY problem coincides with the problem of deciding whether there is a homomorphism from a given graph $\mathbf{G}$ to $\mathbf{K}_3$, where $\mathbf{K}_3$ is the complete graph with three nodes (the triangle). The uniform version of the HOMOMORPHISM PROBLEM, in which both structures $\mathbf{A}$ and $\mathbf{B}$ are given as input, is the most general formulation of the constraint-satisfaction problem. Interesting algorithmic problems, however, also arise by fixing the structure $\mathbf{B}$, which sometimes is called the *template structure*. Thus, the resulting problem, denoted by $\mathrm{CSP}(\mathbf{B})$, asks: given $\mathbf{A}$, is there a homomorphism from $\mathbf{A}$ to $\mathbf{B}$? Note that $\mathrm{CSP}(\mathbf{K}_3)$ is precisely the 3-COLORABILITY problem; more generally, $\mathrm{CSP}(\mathbf{K}_k)$ is the k-COLORABILITY problem, where $\mathbf{K}_k$ is the complete graph with k-nodes, $k \geq 2$.

With constraint-satisfaction problems presented as homomorphism problems in a unfying way, we are closer to our first goal of defining canonical proof systems. The approach we take is via yet another interpretation of CSPs, this time in terms of database theory, building upon the homomorphism framework. As pointed out in [GJC94], every constraint can be thought of as a table of a relational database, and the set of solutions to a CSP can be identified with the tuples in the *join* of all constraints. This fruitful connection between CSPs and database theory is explored further in [KV00a]. Now, a CSP instance is unsatisfiable precisely when the join of the constraints is empty. We adopt this approach and define a $\mathrm{CSP}(\mathbf{B})$ *refutation of an instance* $\mathbf{A}$ to be a sequence of constraints ending with the empty constraint, such that every constraint in the sequence is an initial constraint, the join of two previous constraints, the projection of some previous constraint, or the weakening of some previous constraint. Projection and weakening are not strictly necessary, but provide a versatile tool for reducing the complexity of the intermediate constraints. Note that the join is a form of constraint propagation, since it allows us to derive new constraints implied by the previous ones. See the work by Freuder [Fre78] for the first theoretical approach to constraint propagation.

The proof systems obtained this way are sound and complete for constraint satisfaction. We embark on the investigation of their general properties by focussing first on the concept of *refutation width*, which is the maximum arity of the constraints in a refutation. Bounding the arity of the constraints generated during the execution of constraint propagation algorithms has already played a crucial role in the development of the theory of CSPs, as a method to achieve tractability [Fre82,Fre90,DP87]. For example, various concepts of consistency popularized by the AI community rely on it [Dec03]. Following the ideas in [FV98,KV00a,AD03], we are able to show that the minimal refutation width of a $\mathrm{CSP}(\mathbf{B})$ instance $\mathbf{A}$ is characterized by a combinatorial game in-

troduced in the context of finite-model theory. In turn, again following [FV98,KV00a], this leads us naturally to considering the treewidth of the instance as a parameter. As a result, we obtain a deeper understanding and also a purely combinatorial characterization of refutation width.

CSP refutations are perhaps too general to be of practical use. The rules are too general and the constraints, if represented explicitly, may be too large. Hence, we propose a *syntactic* counterpart to general CSP refutations, in which all the constraints are somehow succintly represented. Technically speaking, we consider representation classes for the constraints. Some examples include clauses, linear equalities over a finite field, linear inequalities over the integers, decision trees, decision diagrams, and so on. With this new formalism, CSP proofs become purely syntactical objects, closer to their counterparts in propositional logic. As a case study, we investigate the proof system obtained by using ordered binary decision diagrams (OBDDs) as our representation class for constraints. OBDDs possess many desirable algorithmic properties and have been used successfully in many areas, most notably in formal verification (see [Bry92,BCM$^+$92]). We compare the strength of refutations by OBDDs with other proof systems for propositional logic. We show that OBDD-based refutations polynomially simulate both resolution and the Gaussian calculus; moreover, they are exponentially stronger than either of these systems, even when the weakening rule is not allowed. If we make strong use of weakening, then refutations by OBDDs can polynomially simulate the cutting planes proof system with coefficients written in unary (called CP^* in [BPR97]). In particular, OBBDs provide polynomial-size proofs of the pigeonhole principle. This shows already that refutations by OBDDs can be exponentially stronger than resolution, and even Frege (Hilbert-style) systems with formulas of bounded alternation-depth, because the pigeonhole principle is hard for them [Hak85,Ajt88,BIK$^+$92]. Finally, we observe that for a particular order of the variables, refutations by OBDDs have small communication complexity. By combining this with known techniques about feasible interpolation [IPU94,Kra97], we establish that OBDD-based refutations have polynomial-size monotone interpolants, for a particular order of the variables. This gives exponential lower bounds for a number of examples, including the clique-coloring principle, still for that particular order. Whether the restriction on the order is necessary remains an interesting open problem.

2 Preliminaries

Constraint-satisfaction problems. A *relational vocabulary* σ is a collection of *relation symbols* R, each of a specified *arity*. A σ-structure $\mathbf{A}$ consists of a *universe* A, or *domain*, and for each $R \in \sigma$, an *interpretation* $R^{\mathbf{A}} \subseteq A^r$, where r is the arity of R.

Let $\mathbf{B}$ be a finite σ-structure. We denote by $\mathrm{CSP}(\mathbf{B})$ the class of all finite σ-structures $\mathbf{A}$ such that there is a homomorphism from $\mathbf{A}$ to $\mathbf{B}$. Recall that a homomorphism is a mapping from the universe of $\mathbf{A}$ to the universe of $\mathbf{B}$ that preserves the relations. As mentioned in the introduction, each $\mathrm{CSP}(\mathbf{B})$ is a constraint-satisfaction problem. The structure $\mathbf{B}$ is called the *template structure*. Let us discuss how 3-SAT can be modeled by a particular $\mathrm{CSP}(\mathbf{B})$. This will be of help later in the paper. The relational vocabulary consists of four ternary relation symbols $\{R_0, R_1, R_2, R_3\}$ rep-

resenting all possible types of 3-clauses: clauses with no negations, clauses with one negation, clauses with two negations, and clauses with three negations. The template structure $\mathbf{T}$ has the truth tables of these types of clauses: $R_0^{\mathbf{T}} = \{0,1\}^3 - \{000\}$, $R_1^{\mathbf{T}} = \{0,1\}^3 - \{100\}$, $R_2^{\mathbf{T}} = \{0,1\}^3 - \{110\}$, and $R_3^{\mathbf{T}} = \{0,1\}^3 - \{111\}$. Every 3-CNF formula φ gives rise to a σ-structure $\mathbf{A}_\varphi$ with universe the set of variables of φ and relations encoding the clauses of φ; for instance, $R_1^{\mathbf{A}_\varphi}$ consists of all triples (x, y, z) of variables of φ such that $(\neg x \vee y \vee z)$ is one of the clauses of φ. Thus, $\mathrm{CSP}(\mathbf{T})$ is equivalent to 3-SAT, since φ is satisfiable if and only if there is a homomorphism from $\mathbf{A}_\varphi$ to $\mathbf{T}$.

Pebble games. The *existential k-pebble games* were defined in [KV95,KV00a]. The games are played between two players, the Spoiler and the Duplicator, on two σ-structures $\mathbf{A}$ and $\mathbf{B}$ according to the following rules. Each player has a set of k pebbles numbered $\{1, \ldots, k\}$. In each round of the game, the Spoiler can make one of two different moves: either he places a free pebble on an element of the domain of $\mathbf{A}$, or he removes a pebble from a pebbled element of $\mathbf{A}$. To each move of the Spoiler, the Duplicator must respond by placing her corresponding pebble over an element of $\mathbf{B}$, or removing her corresponding pebble from $\mathbf{B}$, respectively. If the Spoiler reaches a round in which the set of pairs of pebbled elements is not a partial homomorphism between $\mathbf{A}$ and $\mathbf{B}$, then he wins the game. Otherwise, we say that the Duplicator wins the game. The formal definition can be found in [KV95,KV00a], and the close relationship between existential pebble games and constraint-satisfaction problems was discussed at length in [KV00b].

Treewidth. The *treewidth* of a graph can be defined in many different ways [Bod98]. One way is this. The treewidth of a graph $\mathbf{G}$ is the smallest positive integer k such that $\mathbf{G}$ is a subgraph of a k-tree, where a k-tree is defined inductively as follows: the $k+1$-clique $\mathbf{K}_{k+1}$ is a k-tree, and if $\mathbf{G}$ is a k-tree, then the result of adding a new node to $\mathbf{G}$ that is adjacent to exactly the nodes of a k-clique of $\mathbf{G}$ (thus forming a (k+1)-clique) is also a k-tree. The *Gaifman graph* of a structure $\mathbf{A}$ is the graph whose set of nodes is the universe of $\mathbf{A}$, and whose edges relate pairs of elements that appear in some tuple of a relation of $\mathbf{A}$. The *treewidth* of a structure is the treewidth of its Gaifman graph.

3 Proof Systems for CSPs

Notions from Database Theory. A *relation schema* $R(x_1, \ldots, x_k)$ consists of a *relation name* R, and a set of *attribute names* $x_1, \ldots, x_k$. A *database schema* σ is a set of relation schemas. A *relation conforming with* a relation schema $R(x_1, \ldots, x_k)$ is a set of k-tuples. A *database over* a database schema σ is a set of relations conforming with the relation schemas in σ. In other words, a database over σ is a σ-structure, except that the universe of the structure is not made explicit. In the sequel, we often conflate the notation and use the same symbol for both a relation schema and a relation conforming with that schema.

We use $\mathbf{x}$ to denote a tuple of attribute names $(x_1, \ldots, x_k)$ and also to denote the set $\{x_1, \ldots, x_k\}$. It will be clear from context which case it is. Let R be a relation

conforming with the relational schema $R(\mathbf{x})$. Let $\mathbf{y} \subseteq \mathbf{x}$ be a subset of the set of attribute names. The *projection* of R with respect to $\mathbf{y}$ is the relation whose attribute names are $\mathbf{y}$, and whose tuples can be extended to tuples in R. We denote it $\pi_{\mathbf{y}}(R)$. Let R and S be relations conforming with relational schemas $R(\mathbf{x})$ and $S(\mathbf{y})$. The *relational join* of R and S, or simply *join*, is the largest relation T whose attribute names are $\mathbf{x} \cup \mathbf{y}$, and such that $\pi_{\mathbf{x}}(T) \subseteq R$ and $\pi_{\mathbf{y}}(T) \subseteq S$. We denote it by $R \bowtie S$. Joins are commutative and associative, and can be extended to an arbitrary number of relations.

Notions from CSPs. Let σ be a relational vocabulary. Let $\mathbf{A}$ and $\mathbf{B}$ be two σ-structures. A k-ary *constraint* is a pair $(\mathbf{x}, R)$, where $\mathbf{x}$ is a k-tuple of distinct elements of the universe of $\mathbf{A}$, and R is a k-ary relation over the universe of $\mathbf{B}$. The constraint $(\mathbf{x}, R)$ can be interpreted as a pair formed by a relation schema $R(\mathbf{x})$ and a relation R conforming with it. Here, $\mathbf{x}$ is the set of attribute names. Thus, it makes sense to talk about joins and projections of constraints. We say that a constraint $(\mathbf{x}, R)$ is a *superset*, or *weakening*, of another constraint $(\mathbf{y}, S)$ if $\mathbf{x} = \mathbf{y}$ and $R \supseteq S$.

If there is a homomorphism from $\mathbf{A}$ to $\mathbf{B}$, then we say that the instance $\mathbf{A}$ of $\mathrm{CSP}(\mathbf{B})$ is *satisfiable*; otherwise, we say that it is *unsatisfiable*. Recall from Section 2 that these definitions generalize Boolean satisfiability and unsatisfiability of 3-CNF formulas. If a CSP instance is unsatisfiable, its satisfiability may be *refuted*. We are interested in *refutations* by means of joins, projections, and weakening. Here, constraints $(\mathbf{x}, R)$ are viewed as relational schemas $R(\mathbf{x})$ with a relation R conforming with it as suggested in the preceding paragraph.

Definition 1 (CSP Refutation). *Let $\mathbf{A}$ and $\mathbf{B}$ be σ-structures. A $\mathrm{CSP}(\mathbf{B})$ proof from $\mathbf{A}$ is a finite sequence of constraints $(\mathbf{x}, R)$ each of which is of one of the following forms:*

1. *Axiom: $(\mathbf{x}, R^{\mathbf{B}})$, where $R \in \sigma$ and $\mathbf{x} \in R^{\mathbf{A}}$*
2. *Join: $(\mathbf{x} \cup \mathbf{y}, R \bowtie S)$, where $(\mathbf{x}, R)$ and $(\mathbf{y}, S)$ are previous constraints.*
3. *Projection: $(\mathbf{x} - \{x\}, \pi_{\mathbf{x}-\{x\}}(R))$, where $(\mathbf{x}, R)$ is a previous constraint.*
4. *Weakening: $(\mathbf{x}, S)$, where $(\mathbf{x}, R)$ is a previous constraint and $R \subseteq S$.*

A $\mathrm{CSP}(\mathbf{B})$ refutation of $\mathbf{A}$ is a proof whose last constraint has an empty relation.

Note that the projections eliminate one variable at a time. We say that the variable is projected out. The following simple result states that CSP refutations form a sound and complete method for proving that a given instance of a CSP is unsatisfiable. The fact that CSP can be reduced to a join of constraints is mentioned already in [GJC94].

Theorem 1 (Soundness and Completeness). *Let $\mathbf{A}$ and $\mathbf{B}$ be σ-structures. Then, $\mathbf{A}$ has a $\mathrm{CSP}(\mathbf{B})$ refutation if and only if A is unsatisfiable in $\mathrm{CSP}(\mathbf{B})$. In fact, axioms and joins alone are already enough to refute an unsatisfiable instance.*

Due to space limitations, we need to omit most proofs in this version of the paper. The proof of Theorem 1 shows that refutations need not be any longer than linear in the number of constraints of the CSP instance. However, the critical reader may observe

that the intermediate constraints may be arbitrarily complex. On the other hand, the rules of projection and weakening can be used to lower this complexity when necessary. It will become clear later on how this is of any help in applications. At this point, let us introduce a formalism to measure the complexity of the intermediate constraints. A k-ary constraint $(\mathbf{x}, R)$ can be identified with a Boolean-valued function $f : B^k \to \{0, 1\}$ by letting $f(\mathbf{a}) = 1$ if and only if $\mathbf{a} \in R$ (in other words, this is the *characteristic* function of the relation R). Now, functions of this sort can be represented in various ways by means of representation classes.

Definition 2 (Representation Class). *Let B be a finite set. A representation class for Boolean-valued functions with domain B^k is a triple $\mathcal{R} = (Q, I, S)$, where Q is a set, called the set of representations, I is a mapping from Q to the set of functions $f : B^k \to \{0, 1\}$ called the interpretation, and S is a mapping from Q to the integers, called the size function.*

To be useful for CSP refutations, representation classes should satisfy certain regularity conditions, such as being closed under joins and projections. In addition, the size function should capture the intuitive notion of *complexity* of a representation. There are many examples of representation classes in the literature, particularly when the domain B is Boolean, that is, $B = \{0, 1\}$.

Examples. Let $B = \{0, 1\}$, and let $A = \{x_1, \ldots, x_n\}$ be a set of propositional variables. Clauses over A form a representation class. The interpretation of a clause is the obvious one, and we may define the size of a clause by the number of literals in it. A clause C can be thought of as a constraint $(\mathbf{x}, R)$, where $\mathbf{x}$ is the set of variables in C (not literals), and R is the set of truth assignments to the variables that satisfy the clause. Unfortunately, clauses are not closed under joins, that is, the join of two clauses is not necessarily a clause. Nonetheless, clauses are closed under the *resolution rule*, which can be seen as a combination of one join and one projection (see also [DvB97]). Indeed, if $C \vee x$ and $D \vee \neg x$ are clauses, then the *resolvent* clause $C \vee D$ is precisely the result of projecting x out of their join. We exploit and elaborate on this connection with resolution in Section 5. Binary decision diagrams (BDDs), a.k.a. branching programs (BPs), also form a representation class (see section 5 for a reminder of the definitions). The interpretation of a BDD is the obvious one, and we may define its size by the number of nodes of its graph. BDDs are closed under joins and projections. In fact, BDDs are closed under all operations, since BDDs can represent all Boolean functions. Moreover, when an order on the variables is imposed, the representation of the join can be obtained in polynomial time. We will discuss these issues in Section 5. Linear inequalities $\sum_i a_i x_i \leq a_0$, for integers a_i, also form a representation class. The interpretation of $\sum_i a_i x_i \leq a_0$ is a k-ary Boolean-valued function $f : B^k \to \{0, 1\}$, where k is the number of variables, defined by $f(b_1, \ldots, b_k) = 1$ if and only if $\sum_i a_i b_i \leq a_0$. The size of a linear inequality may be defined by the number of bits needed to represent the $k + 1$ coefficients, or by $a_0 + \sum_i a_i$ if the coefficients are represented in unary. As was the case with clauses, linear inequalities are not closed under joins. Representation classes can also be used to represent functions $f : B^k \to \{0, 1\}$ with non-Boolean domain B. As long as B is finite, BDDs form an appropriate example. The particular case of (non-binary) decision trees is also a good example.

The notion of a representation class suggests a syntactic counterpart of the general notion of CSP refutation. Moreover, it also suggests a way to bound the complexity of the intermediate relations in a CSP refutation. Recall that the width of constraint $(\mathbf{x}, R)$ is the same as its arity, that is, the length of the tuple $\mathbf{x}$.

Definition 3 (Complexity Measures). *Let $\mathbf{B}$ be a σ-structure. Let $\mathcal{R} = (Q, I, S)$ be a representation class for Boolean-valued functions on the universe of $\mathbf{B}$. Let $C_1, \ldots, C_m$ be a $\mathrm{CSP}(\mathbf{B})$ proof, and let $r_i \in Q$ be a representation of the constraint C_i. We say that $r_1, \ldots, r_m$ is an $\mathcal{R}$-proof. Its length is m, its size is $S(r_1) + \cdots + S(r_m)$, and its width is the maximum width of $C_1, \ldots, C_m$.*

It was mentioned already that a representation class should satisfy certain regularity conditions. The actual conditions depend on the application at hand. One particularly useful property is that the representation of a join (projection, weakening) be computable in polynomial time from the representations of the given constraints. In our intended applications, this will indeed be the case.

4 Refutation Width and Treewidth

Characterization of refutation width. Width has played a crucial role in the development of the theory of CSPs [DP87]. Part of the interest comes from the fact that a width upper bound translates, for most representations, to a size bound on individual constraints. This is true, for example, for explicit representation and for BDDs. In the proof complexity literature, Ben-Sasson and Wigderson [BSW01] viewed it as a complexity measure for resolution. Here, we adopt the methods for CSP refutations.

Theorem 2. *Let $\mathbf{A}$ and $\mathbf{B}$ be two finite σ-structures. The following are equivalent:*

1. *$\mathbf{A}$ has a $\mathrm{CSP}(\mathbf{B})$ refutation of width k.*
2. *The Spoiler wins the existential k-pebble game on $\mathbf{A}$ and $\mathbf{B}$.*

An intimate connection between pebble games and the notion of strong consistency [Dec92] was established in [KV00b]. This entails an intimate connection between the concepts of refutation width and the concept of strong consistency. Specifically, it follows from the results in [KV00b] and the above theorem that $\mathbf{A}$ has a $\mathrm{CSP}(\mathbf{B})$ refutation of width k precisely when it is impossible to establish strong k-consistency for $\mathbf{A}$ and $\mathbf{B}$.

Next we turn to studying the effect of the treewidth of the instance $\mathbf{A}$ on the width of the CSP refutations. We will need the following result due to Dalmau, Kolaitis and Vardi:

Theorem 3 ([DKV02]). *Let $k \geq 2$, let $\mathbf{A}$ be a finite σ-structure of treewidth less than k, and let $\mathbf{B}$ be a finite σ-structure. Then the following statements are equivalent:*

1. *There is a homomorphism from $\mathbf{A}$ to $\mathbf{B}$.*
2. *The Duplicator wins the existential k-pebble game on $\mathbf{A}$ and $\mathbf{B}$*

It is immediate from Theorems 2 and 3 that if $\mathbf{A}$ is unsatisfiable in $\mathrm{CSP}(\mathbf{B})$ and has treewidth less than k, then $\mathbf{A}$ has a $\mathrm{CSP}(\mathbf{B})$ refutation of width k. In fact, this result remains true in a more general situation. A substructure $\mathbf{C}$ of $\mathbf{A}$ is called a *core* of $\mathbf{A}$ if there is a homomorphism from $\mathbf{A}$ to $\mathbf{C}$, but, for every proper substructure $\mathbf{C}'$ of $\mathbf{C}$, there is no homomorphism from $\mathbf{A}$ to $\mathbf{C}'$. It is known [HN92] that every finite structure $\mathbf{A}$ has a unique core up to isomorphism, denoted by $\mathrm{core}(\mathbf{A})$, and that $\mathbf{A}$ is homomorphically equivalent to $\mathrm{core}(\mathbf{A})$. In the context of database theory, the treewidth of the core of $\mathbf{A}$ captures exactly the smallest number k such that the canonical conjunctive query $Q^{\mathbf{A}}$ can be expressed in the existential positive fragment of first-order logic with k variables [DKV02, Theorem 12]. Now, back to refutations, if $\mathbf{A}$ is an unsatisfiable instance of $\mathrm{CSP}(\mathbf{B})$ and the core of $\mathbf{A}$ has treewidth less than k, then $\mathbf{A}$ also has a $\mathrm{CSP}(\mathbf{B})$ refutation of width k. Indeed, if $\mathbf{A}$ is an unsatisfiable instance of $\mathrm{CSP}(\mathbf{B})$, so is $\mathrm{core}(\mathbf{A})$ because they are homomorphically equivalent; moreover, if $\mathrm{core}(\mathbf{A})$ has treewidth less than k, then $\mathrm{core}(\mathbf{A})$ has a $\mathrm{CSP}(\mathbf{B})$ refutation of width less than k. Since $\mathrm{core}(\mathbf{A})$ is a substructure of $\mathbf{A}$, a $\mathrm{CSP}(\mathbf{B})$ refutation of $\mathrm{core}(\mathbf{A})$ is also a $\mathrm{CSP}(\mathbf{B})$ refutation of $\mathbf{A}$.

One may wonder whether the converse is true. Is the treewidth of the core of $\mathbf{A}$ capturing the width of the refutations of $\mathbf{A}$? Unfortunately, the answer turns out to be negative for rather trivial reasons. Take a $\mathbf{B}$ such that $\mathrm{CSP}(\mathbf{B})$ can be solved by a k-Datalog program for some fixed k. For example, let $\mathbf{B} = \mathbf{K}_2$ so that $\mathrm{CSP}(\mathbf{B})$ becomes 2-COLORABILITY, which is expressible in 3-Datalog. Take a graph $\mathbf{G}$ which is not 2-colorable. Hence, the Spoiler wins the existential 3-pebble game on $\mathbf{G}$ and $\mathbf{K}_2$ [KV00a]. Now just add an arbitrarily large clique to $\mathbf{G}$, that is, let $\mathbf{G}' = \mathbf{G} \cup \mathbf{K}_k$ for some large k. There still exists a $\mathrm{CSP}(\mathbf{B})$ refutation of $\mathbf{G}'$ of width 3, but the core of $\mathbf{G}'$ has treewidth at least $k-1$. This counterexample, however, suggests that something more interesting is going on concerning the relationship between existential k-pebble games and treewidth k.

Theorem 4. *Let $k \geq 2$, let $\mathbf{A}$ and $\mathbf{B}$ be two finite σ-structures. Then the following statements are equivalent:*

1. *The Duplicator wins the existential k-pebble game on $\mathbf{A}$ and $\mathbf{B}$.*
2. *If $\mathbf{A}'$ is a structure of treewidth less than k and such that there is a homomorphism from $\mathbf{A}'$ to $\mathbf{A}$, then the Duplicator wins the existential k-pebble game on $\mathbf{A}'$ and $\mathbf{B}$.*

Proof sketch: (i) $\Rightarrow$ (ii) is easy. (ii) $\Rightarrow$ (i). Let $P_{\mathbf{B}}$ be the k-Datalog program that expresses the query: "Given $\mathbf{A}$, does the Spoiler win the existential k-pebble game on $\mathbf{A}$ and $\mathbf{B}$?" [KV00a]. Assume that the Spoiler wins the existential k-pebble game on $\mathbf{A}$ and $\mathbf{B}$. Hence $\mathbf{A}$ satisfies $P_{\mathbf{B}}$, hence it satisfies one of the stages of the k-Datalog program $P_{\mathbf{B}}$. Each such stage is definable by a union of conjunctive queries, each of which can be written in the existential positive fragment of first-order logic with k variables. Hence $\mathbf{A}$ satisfies $Q^{\mathbf{A}'}$, where $\mathbf{A}'$ is a structure of treewidth less than k. Hence, there is a homomorphism h from $\mathbf{A}'$ to $\mathbf{A}$. But also $\mathbf{A}'$ satisfies $P_{\mathbf{B}}$, hence the Spoiler wins the existential k-pebble game on $\mathbf{A}'$ and $\mathbf{B}$. □

Now we combine Theorems 2, 3 and 4 to obtain a purely combinatorial characterization of when a structure has a CSP refutation of a certain width.

Corollary 1. *Let $k \geq 2$, let $\mathbf{A}$ and $\mathbf{B}$ be two finite σ-structures. The following are equivalent:*

1. *$\mathbf{A}$ has a $\mathrm{CSP}(\mathbf{B})$ refutation of width k.*
2. *There exists a structure $\mathbf{A}'$ of treewidth less than k and such that there is a homomorphism from $\mathbf{A}'$ to $\mathbf{A}$, and $\mathbf{A}'$ is unsatisfiable in $\mathrm{CSP}(\mathbf{B})$.*

Note that the characterization of refutation width is stated in terms of treewidth and homomorphisms and does not mention refutations at all. Let us add that the structure in (2) can be large, so Corollary 1 does not yield any complexity bound for deciding whether $\mathbf{A}$ has a $\mathrm{CSP}(\mathbf{B})$ refutation of width k. As it turns out, it follows from Theorem 2 and the result in [KP03], that this problem is EXPTIME-complete.

Small-width proof-search algorithms. Next we study the complexity of finding a satisfying assignment, or refuting the satisfiability, of an instance $\mathbf{A}$ of $\mathrm{CSP}(\mathbf{B})$ when we parameterize by the treewidth k of $\mathbf{A}$. The decision problem has been studied before in certain particular cases. When k is bounded by a constant, the problem can be solved in polynomial time [DP87,Fre90]. When $\mathbf{B}$ is a fixed structure, Courcelle's Theorem [Cou90] implies that the problem can be solved in time $2^{O(k)}n$, where n is the size of $\mathbf{A}$. Indeed, if $\mathbf{B}$ is fixed, then satisfiability in $\mathrm{CSP}(\mathbf{B})$ can be expressed in monadic second-order logic, so Courcelle's Theorem applies. We consider the case in which $\mathbf{B}$ and k are not fixed, and also the problem of finding a satisfying assignment, or a refutation. In the particular case of Boolean $\mathbf{B}$ and resolution refutations, a related problem was studied in [AR02] where branchwidth was used instead of treewidth. Our proof is more general, rather different, and perhaps simpler.

Theorem 5. *The problem of determining whether a structure $\mathbf{A}$ of treewidth k is satisfiable in $\mathrm{CSP}(\mathbf{B})$ can be solved by a deterministic algorithm in time $2^{O(k)}m^{O(k)}n^{O(1)}$, where n is the size of $\mathbf{A}$ and m is the size of $\mathbf{B}$. In particular, the algorithm runs in polynomial time when $k = O(\log n / \log m)$. Moreover, if $\mathbf{A}$ is satisfiable, the algorithm produces a homomorphism $h : \mathbf{A} \to \mathbf{B}$, and if $\mathbf{A}$ is unsatisfiable, it produces a $\mathrm{CSP}(\mathbf{B})$ refutation of width k.*

Proof sketch. The idea is to build an existential positive sentence ψ, with k variables, that is a rewriting of the canonical query $Q^{\mathbf{A}}$. This takes time polynomial in the tree-decomposition of $\mathbf{A}$, which can be found in time $2^{O(k)}n^{O(1)}$. Then we evaluate ψ on $\mathbf{B}$ bottom up, from inner subformulas to the root. Since each subformula involves at most k variables, this takes time $m^{O(k)}$ times the size of the formula, which is time $2^{O(k)}m^{O(k)}n^{O(1)}$ overall. Since $\psi \equiv Q^{\mathbf{A}}$, we have that $\mathbf{B}$ satisfies ψ if and only if there exists a homomorphism from $\mathbf{A}$ to $\mathbf{B}$. □

5 Refutations by OBDDs: a case study

Regularity properties of OBDDs. In this section we study the effect of using *ordered binary decision diagrams* as a representation class for constraints. We focus on the Boolean case $B = \{0, 1\}$.

For the history on the origins of binary decision diagrams, branching programs, and ordered binary decision diagrams we refer the reader to the survey by Bryant [Bry92]. Here are the definitions. Let $x_1, \ldots, x_n$ be n propositional variables. A *binary decision diagram* (BDD), or *branching program* (BP), represents a Boolean function as a rooted, directed acyclic graph $\mathbf{G}$. Each non-terminal node u of $\mathbf{G}$ is labeled by a variable $v(u) \in \{x_1, \ldots, x_n\}$, and has arcs toward two children $t(u)$ and $f(u)$, referred to as the true and the false children respectively. Each terminal node is labeled 0 or 1. For a truth assigment to the variables $x_1, \ldots, x_n$, the value of the function is determined by following the path through the directed graph, from the root to a terminal node, according to the labels of the nodes and the values to the variables. The size of a BDD is the size of the underlying graph $\mathbf{G}$. An *ordered binary decision diagram* (OBDD) is a BDD in which labeled paths are consistent with a specific total order $<$ over the variables. More precisely, for an OBDD we require that the variable labeling a non-terminal node be smaller than the variables labeling its non-terminal children, according to a fixed order over the variables.

The main property of OBDDs is that, in their reduced form, they are *canonical*, meaning that for a given order, two OBDDs for the same function are isomorphic. An immediate consequence is that testing for equivalence of two OBDDs can be solved in time polynomial in their size. Most interesting for us is the fact that representations of joins and projections are computable in polynomial time, and determining whether an OBDD is a weakening of another is decidable in polynomial time.

It follows from this that given a CSP refutation $C_1, \ldots, C_m$ with the constraints represented by OBDDs, the validity of applications of the join rule, the projection rule, and the weakening rule, can be checked in polynomial time. Therefore, refutations by OBDDs when applied to 3-SAT (a particular $\mathrm{CSP}(\mathbf{B})$, see below) form a proof system in the sense of Cook and Reckhow [CR79].

Strength of refutations by OBDDs. Let us compare the size of CSP refutations by OBDDs with other well-known proof systems for propositional logic. Recall from Section 2 how 3-SAT is represented as a $\mathrm{CSP}(\mathbf{B})$ problem. The template structure is $\mathbf{T}$ and its vocabulary consists of four ternary relations $\{R_0, R_1, R_2, R_3\}$, one for each type of 3-clause. Thus, structures for this vocabulary are 3-CNF formulas. A *refutation by OBDDs* of a 3-CNF formula $\mathbf{A}$ is a refutation of $\mathbf{A}$ in $\mathrm{CSP}(\mathbf{T})$ when constraints are represented by OBDDs for a fixed total order of the variables. Size, length and width of refutations by OBDDs are defined according to Definition 3 in Section 3.

Resolution. The resolution rule is very simple: from $C \vee x$ and $D \vee \neg x$, derive $C \vee D$, where C and D are clauses in which x does not occur. The goal is to derive the empty clause from a given set of initial clauses. The length of a resolution refutation is the number of clauses that are used in it. The size of a resolution refutation is the total number of literals that appear in it. There are two key observations that concern us here. The first is that every clause has a small equivalent OBDD over all orders of the variables. The second observation is that $C \vee D$ can be expressed in terms of one join and one projection from $C \vee x$ and $D \vee \neg x$ (see also [DvB97]). We use both facts for the following result whose proof will be included in the full paper.

Theorem 6. *Let* $\mathbf{A}$ *be a 3-CNF formula on* n *variables. If* $\mathbf{A}$ *has a resolution refutation of length* m*, then* $\mathbf{A}$ *has a refutation by OBDDs of length* $2m$ *and size* $O(mn^2)$*, even without using the weakening rule and for every order of the variables. Moreover, there is a polynomial-time algorithm that converts the resolution refutation into the refutation by OBDDs.*

We will see below that, in fact, refutations by OBDDs are exponentially stronger than resolution. As an intermediate step we move to a different CSP: systems of equations over $\mathbf{Z}_2$.

Gaussian calculus. One nice feature of OBDDs is that they give a uniform framework for defining all types of constraints. Consider now the CSP defined by systems of linear equations over the two-element field $\mathbf{Z}_2$, with exactly three variables per equation. That is, the vocabulary contains two ternary relation symbols R_0 and R_1 representing the equations $x + y + z = 0$ and $x + y + z = 1$ respectively. The template structure $\mathbf{S}$ contains the truth tables of these equations: that is $R_0^{\mathbf{S}} = \{000, 011, 110, 101\}$ and $R_1^{\mathbf{S}} = \{001, 010, 100, 111\}$. Now $\mathrm{CSP}(\mathbf{S})$ coincides with systems of equations over $\mathbf{Z}_2$. The standard method for solving systems of equations is Gaussian elimination. In fact, Gaussian elimination can be used to refute the satisfiability of systems of equations by deriving, for example, $0 = 1$ by means of linear combinations that cancel at least one variable. This has led to proposing the Gaussian calculus as a proof system [BSI99]. Let us see that refutations by OBDDs can polynomially simulate it. Perhaps the most interesting point of the proof is that we actually show that weakening is not required, which is not immediately obvious.

Theorem 7. *Let* $\mathbf{A}$ *be a system of equations over* $\mathbf{Z}_2$ *with exactly three variables per equation. If* $\mathbf{A}$ *has a Gaussian calculus refutation of length* m*, then* $\mathbf{A}$ *has a refutation by OBDDs in* $\mathrm{CSP}(\mathbf{S})$ *of length* $2m$ *and size* $O(mn^2)$*, even without using the weakening rule and for every order of the variables. Moreover, there is a polynomial-time algorithm that converts the Gaussian calculus refutation into the refutation by OBDDs.*

We can now use this result to conclude that for 3-CNF formulas, refutations by OBDDs are exponentially stronger than resolution. Consider the standard translation of a linear equation $x + y + z = a$ of $\mathbf{Z}_2$ into a 3-CNF formula. Namely, for $a = 1$ the 3-CNF formula is

$$(x \vee y \vee z) \wedge (x \vee \neg y \vee \neg z) \wedge (\neg x \vee y \vee \neg z) \wedge (\neg x \vee \neg y \vee z),$$

and the formula for $a = 0$ is similar. For a system of equations over $\mathbf{Z}_2$ with three variables per equation $\mathbf{A}$, let $\mathbf{T}(\mathbf{A})$ be its translation to a 3-CNF formula. It is not hard to see that if $\mathbf{A}$ has a refutation by OBDDs in $\mathrm{CSP}(\mathbf{S})$ of length m, then $\mathbf{T}(\mathbf{A})$ has a refutation by OBDDs in $\mathrm{CSP}(\mathbf{T})$ of length $O(m)$. The idea is that the join of the OBDDs for the clauses defining an equation $x + y + z = a$ is precisely an OBDD representing the equation $x + y + z = a$. Therefore, one refutation reduces to the other.

The particular system of equations known as *Tseitin contradictions* [Tse68] is exponentially hard for resolution. This was shown by Urquhart [Urq87] and was later extended by Ben-Sasson [BS02] who showed the same result for every Frege system (Hilbert-style system) restricted to formulas of bounded alternation-depth. This

establishes that refutations by OBDDs are exponentially stronger than resolution. For bounded alternation-depth Frege systems, it shows that in some cases refutations by OBDDs might be exponentially stronger. In the next section we see that refutations by OBDDs and bounded alternation-depth Frege systems are incomparable.

Cutting planes. We now show that, in the presence of weakening, refutations by OBDDs polynomially simulate the cutting planes proof system with small coefficients. It is well-known that clauses can be expressed as linear inequalities over the integers. For example the clause $x \vee \neg y \vee z$ can be expressed by $x + (1 - y) + z \geq 1$, or equivalently, $x - y + z \geq 0$. Therefore, a CNF formula translates into a system of inequalities over the integers in a natural way. The cutting planes proof system was introduced in [CCT87]. The lines in the proof are linear inequalities over the integers. There are three rules of inference: addition, scalar multiplication, and integer division. The only rule the requires explanation is the integer division. From $\sum_i (c \cdot a_i) x_i \geq a_0$ derive $\sum_i a_i x_i \geq \lceil a_0/c \rceil$. Intuitively, if all coefficients except the independent term are divisible by c, then we may divide all over, and round-up the independent term. The rule is sound on the integers, meaning that if the x_i's take integer values that satisfy the hypothesis, then the conclusion is also satisfied. The goal of the system is to derive a contradiction $0 \geq 1$ from a given set of linear inequalities. For refuting 3-CNF formulas, each clause is viewed as a linear inequality as described before.

In order to measure the size of a proof we need to specify an encoding for the inequalities. When the coefficients are encoded in unary, the system has been named CP^* and studied in [BPR97]. We see that refutations by OBDDs can polynomially simulate CP^*. As it turns out, the rule of weakening is strongly used here. Whether weakening is strictly necessary remains as an intriguing open problem.

Theorem 8. *Let* $\mathbf{A}$ *be a 3-CNF. If* $\mathbf{A}$ *has a* CP^* *refutation of length* m *and size* s*, then* $\mathbf{A}$ *has a refutation by OBDDs of length* $2m$ *and size* $s^{O(1)}$*, for every order of the variables. Moreover, there is a polynomial-time algorithm that converts the* CP^* *refutation into the refutation by OBDDs.*

One consequence of this is that the *pigeonhole principle*, when encoded propositionally as an unsatisfiable 3-CNF formula, admits polynomial-size OBDD refutations. This follows from the known polynomial-size proofs of the pigeonhole principle in cutting planes [CCT87]. In contrast, the pigeonhole principle requires exponential-size refutations in resolution [Hak85]. It would be good to find a direct construction of the polynomial-size OBDD proof of the pigeonhole principle.

Interpolation. Craig's Interpolation Theorem in the propositional setting is this. Let $A(\mathbf{x}, \mathbf{y})$ and $B(\mathbf{y}, \mathbf{z})$ be propositional formulas for which $\mathbf{x}$, $\mathbf{y}$ and $\mathbf{z}$ are pairwise disjoint. If $A(\mathbf{x}, \mathbf{y}) \wedge B(\mathbf{y}, \mathbf{z})$ is unsatisfiable, then there exists a formula $C(\mathbf{y})$ such that $A(\mathbf{x}, \mathbf{y}) \wedge \neg C(\mathbf{y})$ and $C(\mathbf{y}) \wedge B(\mathbf{y}, \mathbf{z})$ are both unsatisfiable. The promised $C(\mathbf{y})$ is called an *interpolant*.

Interpolation has been used in propositional proof complexity as a method for lower bounds. Following earlier working starting in [IPU94,BPR97], Krajícek [Kra97] suggested the following approach. Suppose we are given a refutation of $A(\mathbf{x}, \mathbf{y}) \wedge B(\mathbf{y}, \mathbf{z})$.

Suppose, further, that we are able to extract an interpolant $C(\mathbf{y})$ by manipulation from the proof. Then, lower bounds for the complexity of the interpolants give lower bounds for the refutations. This idea has been used successfully for a number of proof systems including resolution and cutting planes (see [IPU94,BPR97,Kra97,Pud97]). The feasible interpolation of resolution has been recently used by McMillan [McM03] as an effective abstraction technique in symbolic model checking.

Our aim is to discuss the fact that refutations by OBDDs have feasible interpolation for certain orders of the variables. Following the machinery developed in [IPU94], it is enough to observe that evaluating an OBDD requires small communication complexity for nice orders. We omit further details in this version and state the final result without proof. The *narrowness* of an OBDD is the maximum number of nodes in a level.

Theorem 9. *Let $\mathbf{F} = A(\mathbf{x},\mathbf{y}) \wedge B(\mathbf{y},\mathbf{z})$ be an unsatisfiable 3-CNF formula, and let $n = |\mathbf{y}|$. If $\mathbf{F}$ has an OBDD refutation of length m with OBDDs of narrowness bounded by c, and with an order that is consistent with $\mathbf{x} < \mathbf{y} < \mathbf{z}$, then $\mathbf{F}$ has an interpolant circuit of size $O(c^2(m+n))$. In particular, if the size of the refutation is s, then the size of the interpolant is $s^{O(1)}$. In addition, if $A(\mathbf{x},\mathbf{y})$ is monotone in $\mathbf{y}$, then the interpolant circuit is monotone.*

Let us mention that the monotone feasible interpolation of refutations by OBDDs establishes a separation from Frege systems with formulas of bounded alternation-depth. It is known that monotone interpolants for such systems require exponential-size [Kra97]. This, together with the results of previous sections, establishes that refutations by OBDDs are incomparable in strength with Frege systems of bounded alternation-depth.

6 Concluding Remarks

Viewing constraint propagation as a proof system lifts proof complexity from propositional logic to all constraint-satisfaction problems. There are many questions that remain open from our work.

First, it is necessary to have better understanding of the role of the weakening rule. We know it is not needed to achieve completeness, not even in the case of restricted refutation width in Theorem 2. It remains an open problem to determine whether refutation by OBDDs without weakening can polynomially simulate CP^* refutations. Clarifying the role of weakening is also important for algorithmic applications. Second, the proof complexity of refutations by OBDDs needs further development. One problem that is left open is to find a non-trivial lower bound for the size of refutations by OBDDs that holds for every order of the variables. Another problem that is left open is whether OBDD-based refutations polynomially simulate cutting planes with coefficients written in binary. Are OBDD-based refutations automatizable in the sense of [BPR00]? Can we use the feasible interpolation of OBDD-based refutations in an effective manner analogous to that of McMillan [McM03]?

Finally, it would be good to find practical decision procedures based on CSP proofs, the same way that the DPLL approach is based on resolution. Some progress in this

direction is reported in [DR94], which reports on SAT-solving using directional resolution, and in [PV04], which reports on SAT-solving using OBDD-based refutations. This could lead to CSP-solvers that deal directly with the CSP instances, avoiding the need to translate to a propositional formula and using a SAT-solver as it is sometimes done.

References

[AD03] A. Atserias and V. Dalmau. A combinatorial characterization of resolution width. In *18th IEEE Conference on Computational Complexity*, pages 239–247, 2003.

[Ajt88] M. Ajtai. The complexity of the pigeonhole principle. In *29th Annual IEEE Symposium on Foundations of Computer Science*, pages 346–355, 1988.

[AR02] M. Alekhnovich and A. Razborov. Satisfiability, branch-width and Tseitin tautologies. In *43rd Annual IEEE Symposium on Foundations of Computer Science*, pages 593–603, 2002.

[BCM+92] J.R. Burch, E.M. Clarke, K.L. McMillan, D.L. Dill, and L.J. Hwang. Symbolic model checking: 10^{20} states and beyond. *Information and Computation*, 98(2):142–170, 1992.

[BIK+92] P. Beame, R. Impagliazzo, J. Krajícek, T. Pitassi, P. Pudlák, and A. Woods. Exponential lower bounds for the pigeonhole principle. In *24th Annual ACM Symposium on the Theory of Computing*, pages 200–220, 1992.

[Bod98] H. L. Bodlaender. A partial k-arboretum of graphs with bounded treewidth. *Theoretical Computer Science*, 209:1–45, 1998.

[BPR97] M. L. Bonet, T. Pitassi, and R. Raz. Lower bounds for cutting planes proofs with small coefficients. *Journal of Symbolic Logic*, 62(3):708–728, 1997.

[BPR00] M. L. Bonet, T. Pitassi, and R. Raz. On interpolation and automatization for Frege systems. *SIAM Journal of Computing*, 29(6):1939–1967, 2000.

[Bry92] R. E. Bryant. Symbolic Boolean manipulation with ordered binary-decision diagrams. *ACM Computing Surveys*, 24(3):293–318, 1992.

[BS02] E. Ben-Sasson. Hard examples for bounded depth frege. In *34th Annual ACM Symposium on the Theory of Computing*, pages 563–572, 2002.

[BSI99] E. Ben-Sasson and R. Impagliazzo. Random CNF's are hard for the polynomial calculus. In *40th Annual IEEE Symposium on Foundations of Computer Science*, pages 415–421, 1999.

[BSW01] E. Ben-Sasson and A. Wigderson. Short proofs are narrow–resolution made simple. *Journal of the ACM*, 48(2):149–169, 2001.

[CCT87] W. Cook, C. R. Coullard, and G. Turán. On the complexity of cutting-plane proofs. *Discrete Applied Mathematics*, 18:25–38, 1987.

[Cou90] B. Courcelle. Graph rewriting: An algebraic and logic approach. In J. van Leeuwen, editor, *Hankbook of Theoretical Computer Science*, volume 2, pages 194–242. Elsevier Science Publishers, 1990.

[CR79] S. Cook and R. Reckhow. The relative efficiency of propositional proof systems. *Journal of Symbolic Logic*, 44:36–50, 1979.

[Dec92] R. Dechter. From local to global consistency. *Artificial Intelligence*, 55(1):87–107, May 1992.

[Dec03] R. Dechter. *Constraint Processing*. Morgan Kaufmman, 2003.

[DKV02] V. Dalmau, Ph. G. Kolaitis, and M. Y. Vardi. Constraint satisfaction, bounded treewidth, and finite variable logics. In *8th International Conference on Principles and Practice of Constraint Programming (CP),* volume 2470 of *Lecture Notes in Computer Science*, pages 310–326. Springer, 2002.

[DP87] R. Dechter and J. Pearl. Network-based heuristics for constraint-satisfaction problems. *Artificial Intelligence*, 34:1–38, 1987.

[DR94] R. Dechter and I. Rish. Directional Resolution: The Davis-Putnam Procedure, Revisited. In *4th International Conference on Principles of Knowledge Representation and Reasoning (KR)*, pages 134–145. Morgan Kaufmann, 1994.

[DvB97] R. Dechter and P. van Beek. Local and global relational consistency. *Theoretical Computer Science*, 173(1):283–308, 1997.

[Fre78] E. C. Freuder. Synthesizing constraint expressions. *Communications of the ACM*, 21(11):958–966, 1978.

[Fre82] E. C. Freuder. A sufficient condition for backtrack-free search. *Journal of the Association for Computing Machinery*, 29(1):24–32, 1982.

[Fre90] E. C. Freuder. Complexity of k-tree structured constraint satisfaction problems. In *Proc. AAAI-90*, pages 4–9, 1990.

[FV98] T. Feder and M. Y. Vardi. The computational structure of monotone monadic SNP and constraint satisfaction: A study through Datalog and group theory. *SIAM Journal of Computing*, 28(1):57–104, 1998.

[GJC94] M. Gyssens, P.G. Jeavons, and D.A. Cohen. Decomposition constraint satisfaction problems using database techniques. *Artificial Intelligence*, 66:57–89, 1994.

[Hak85] A. Haken. The intractability of resolution. *Theoretical Computer Science*, 39:297–308, 1985.

[HN92] P. Hell and J. Nešetřil. The core of a graph. *Discrete Mathematics*, 109:117–126, 1992.

[IPU94] R. Impagliazzo, T. Pitassi, and A. Urquhart. Upper and lower bounds for tree-like cutting planes proofs. In *9th IEEE Symposium on Logic in Computer Science*, pages 220–228, 1994.

[KP03] Ph. G. Kolaitis and J. Panttaja. On the complexity of existential pebble games. In *Computer Science Logic '2003, 17th Annual Conference of the EACSL*, volume 2803 of *Lecture Notes in Computer Science*, pages 314–329. Springer, 2003.

[Kra97] J. Krajícek. Interpolation theorems, lower bounds for proof systems, and independence results for bounded arithmetic. *Journal of Symbolic Logic*, 62:457–486, 1997.

[KV95] Ph. G. Kolaitis and M. Y. Vardi. On the expressive power of Datalog: tools and a case study. *Journal of Computer and System Sciences*, 51:110–134, 1995.

[KV00a] Ph. G. Kolaitis and M. Y. Vardi. Conjunctive-query containment and constraint satisfaction. *Journal of Computer and System Sciences*, 61(2):302–332, 2000.

[KV00b] Ph. G. Kolaitis and M. Y. Vardi. A game-theoretic approach to constraint satisfaction. In *17th National Conference on Artificial Intelligence*, pages 175–181, 2000.

[McM03] K. L. McMillan. Interpolation and SAT-based model checking. In Warren A. Hunt Jr. and Fabio Somenzi, editors, *Proceedings of the 15th International Conference on Computer Aided Verification (CAV)*, volume 2725 of *Lecture Notes in Computer Science*, pages 1–13. Springer, 2003.

[Pud97] P. Pudlák. Lower bounds for resolution and cutting plane proofs and monotone computations. *Journal of Symbolic Logic*, 62(3):981–998, 1997.

[PV04] G. Pan and M. Y. Vardi. Search vs. symbolic techniques in satisfiability solving. To appear in Proceedings 7th International Conference on Theory and Applications of Satisfiability Testing, 2004.

[Tse68] G. S. Tseitin. *On the complexity of derivation in propositional calculus*, pages 115–125. Consultants Bureau, 1968.

[Urq87] A. Urquhart. Hard examples for resolution. *Journal of the ACM*, 34(1):209–219, 1987.