Hybrid tractable CSPs which generalize tree structure

Martin C. Cooper[1] **and Peter G. Jeavons**[2] **and András Z. Salamon**[3]

Abstract. The constraint satisfaction problem (CSP) is a central generic problem in artificial intelligence. Considerable progress has been made in identifying properties which ensure tractability in such problems, such as the property of being tree-structured. In this paper we introduce the broken-triangle property, which allows us to define a hybrid tractable class for this problem which significantly generalizes the class of problems with tree structure. We show that the broken-triangle property is conservative (i.e., it is preserved under domain reduction and hence under arc consistency operations) and that there is a polynomial-time algorithm to determine an ordering of the variables for which the broken-triangle property holds (or to determine that no such ordering exists). We also present a non-conservative extension of the broken-triangle property which is also sufficient to ensure tractability and can be detected in polynomial time.

Keywords: constraint satisfaction, tractability, computational complexity, arc consistency.

1 INTRODUCTION

Constraint satisfaction problems with tree structure have been widely studied, and are known to have efficient algorithms [8].

However, tree structure is quite restricted. It is therefore worthwhile exploring more general problem classes, to identify more widely-applicable properties which still allow efficient solution algorithms. A subclass of the general CSP which can be solved in polynomial time, and also identified in polynomial time, is called a *tractable* subclass.

There has been a considerable research effort in identifying tractable subclasses of the CSP over the past decade. Most of this work has focused on one of two general approaches: either identifying forms of constraint which are sufficiently restrictive to ensure tractability no matter how they are combined [2, 9], or else identifying structural properties of constraint networks which ensure tractability no matter what forms of constraint are imposed [7, 5].

The first approach has had considerable success in characterizing precisely which forms of constraint ensure tractability no matter how they are combined. A set of constraint types with this property is called a tractable *constraint language*. In general it has been shown that *any* tractable constraint language must have certain algebraic properties known as polymorphisms [13]. A complete characterization of all possible tractable constraint languages has been established in the following cases: *conservative* constraint languages (i.e. constraint languages containing all unary constraints) [3], and constraint languages over a 2-element domain [17] or a 3-element domain [4].

The second approach has also had considerable success in characterizing precisely which structures of constraint network ensure tractability no matter what constraints are imposed. For the class of problems where the arity of the constraints is bounded by some fixed constant (such as binary constraint problems) it has been shown that (subject to certain technical assumptions) the *only* class of structures which ensure tractability are structures of *bounded tree-width* [12].

However, many constraint satisfaction problems do not possess a sufficiently restricted structure or use a sufficiently restricted constraint language to fall into any of these tractable classes. They may still have properties which ensure they can be solved efficiently, but these properties concern both the structure and the form of the constraints. Such properties have sometimes been called *hybrid* reasons for tractability [16], and they are much less widely-studied and much less well-understood than the language properties and structural properties described above.

In this paper we introduce a new hybrid property which we call the *broken-triangle property*. We show that this property is sufficient to ensure that a CSP instance is tractable, and also show that checking whether an instance has the broken-triangle property can be done in polynomial time. Moreover, we show that all tree-structured CSP instances have this property, as well as many other instances that are not tree-structured (including some with unbounded tree-width).

The broken triangle property can be thought of as a kind of transitivity condition. By processing the variables in an appropriate order, an algorithm akin to those used for solving tree-structured CSP instances can be applied to find a solution. Moreover, a suitable such ordering of variables can be found efficiently. The general technique for finding a suitable ordering, and then exploiting it to generate a solution, is discussed in Section 3. Sections 4 to 6 extend these ideas.

2 THE BROKEN TRIANGLE PROPERTY

In this paper we focus on binary constraint satisfaction problems. A **binary relation** over domains D_i and D_j is a subset of $D_i \times D_j$. For a binary relation R, the relation $\mathrm{rev}(R)$ is defined as $\{(v,u) \mid (u,v) \in R\}$.

A binary CSP **instance** consists of a set of **variables** (where each variable is denoted by a number $i \in \{1, 2, \ldots, n\}$); for each variable i, a **domain** D_i containing possible *values* for variable i; and a set of **constraints**, each of the form $\langle (i,j), R \rangle$, where i and j are variables and R is a relation such that $R \subseteq D_i \times D_j$.

To simplify the notation we introduce the notion of a *canonical constraint relation* which combines all of the specified information about a pair of variables i, j.

Definition 1 *Suppose i and j are variables of a CSP instance. De-*

[1] IRIT, University of Toulouse III, 31062 Toulouse, France, email: cooper@irit.fr
[2] Computing Laboratory, University of Oxford, Oxford, OX1 3QD, UK, email: Peter.Jeavons@comlab.ox.ac.uk
[3] Computing Laboratory, University of Oxford, Oxford, OX1 3QD, UK, and The Oxford-Man Institute of Quantitative Finance, 9 Alfred Street, Oxford, OX1 4EH, UK, email: Andras.Salamon@comlab.ox.ac.uk

note by U_{ij} the set of constraint relations specified for the (ordered) pair of variables (i, j). The **canonical constraint relation** *between variables i and j will be denoted R_{ij} and is defined as*

$$R_{ij} = \bigcap (U_{ij} \cup \{\mathrm{rev}(R) \mid R \in U_{ji}\}).$$

The canonical constraint relation R_{ij} contains precisely the pairs of values that are allowed for the variables i and j by all the constraints on i and j. Note that $R_{ij} = \mathrm{rev}(R_{ji})$. If there are no constraints involving i and j, then R_{ij} is the intersection of an empty set, and is defined to be the *complete* relation $D_i \times D_j$. If relation R_{ij} is neither empty nor the complete relation, we say it is *proper*.

Definition 2 *A binary CSP instance satisfies the* **broken-triangle property (BTP)** *with respect to (w.r.t.) the variable ordering $<$, if, for all triples of variables i, j, k such that $i < j < k$, if $(u, v) \in R_{ij}$, $(u, a) \in R_{ik}$ and $(v, b) \in R_{jk}$, then either $(u, b) \in R_{ik}$ or $(v, a) \in R_{jk}$.*

The broken-triangle property can be understood by the implication shown in Figure 1. In this figure, each oval represents the domain of an associated variable, and each line represents a consistent assignment of values for a pair of variables. A line joins element $u \in D_i$ and element $v \in D_j$ if $(u, v) \in R_{ij}$. The BTP on i, j, k simply says that for any "broken triangle" $a-u-v-b$, as illustrated in Figure 1, there is always a true triangle $u - v - c$ (where c is either a or b). BTP is similar to but stronger than directional path consistency [18].

It is important to note that the BTP must be satisfied for all triples $i < j < k$, even if the description of the instance does not specify a constraint between variables i and j. If there is no specified constraint between i and j, then R_{ij} allows all pairs of values.

A set of CSP instances may satisfy the broken-triangle property due to the structure of the constraint graph, due to the language of the constraint relations, or due to a combination of these.

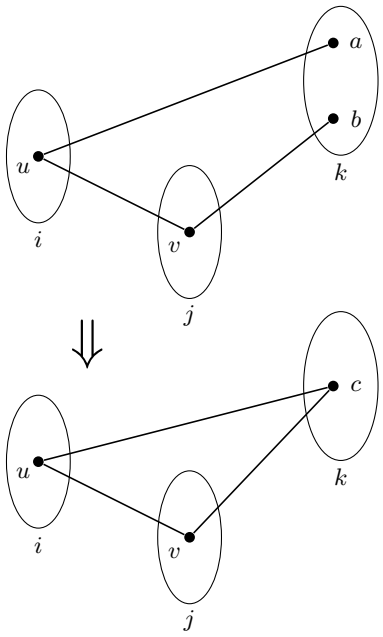

Figure 1. The broken-triangle property on variables i, j, k.

For an element $a \in D_i$, we write $R_{ij}(a)$ to represent $\{b \in D_j : (a, b) \in R_{ij}\}$, the **image** of a in relation R_{ij}.

Lemma 3 *A binary CSP instance satisfies the broken-triangle property with respect to the variable ordering $<$ if and only if for all triples of variables $i < j < k$, and for all $(u, v) \in R_{ij}$,*

$$(R_{ik}(u) \subseteq R_{jk}(v)) \vee (R_{jk}(v) \subseteq R_{ik}(u)). \quad (1)$$

Proof: The condition that either $R_{ik}(u) \subseteq R_{jk}(v)$ or $R_{jk}(v) \subseteq R_{ik}(u)$ is equivalent to stating that there do not exist elements a of $R_{ik}(u)$ and b of $R_{jk}(v)$ such that $a \notin R_{jk}(v)$ and $b \notin R_{ik}(u)$. By the definition of the image of an element in a relation, this in turn is equivalent to the statement that there do not exist $a, b \in D_k$ such that $(u, a) \in R_{ik}$, $(v, b) \in R_{jk}$, $(u, b) \notin R_{ik}$ and $(v, a) \notin R_{jk}$. Sentence (1) therefore exactly forbids the presence of a configuration that would prevent the instance from satisfying the BTP. ∎

Using this result we can obtain the following simple sufficient condition for the broken-triangle property.

Lemma 4 *A binary CSP instance satisfies the broken-triangle property with respect to a variable ordering $<$ if, for all triples of variables $i < j < k$, either R_{ik} or R_{jk} is a complete relation.*

Proof: If R_{ik} is a complete relation, then $R_{ik}(u) = D_k$, while if R_{jk} is a complete relation, then $R_{jk}(v) = D_k$. In either case, by Lemma 3, the instance satisfies the BTP. ∎

Definition 5 *A class of CSP instances is called* **conservative** *if it is closed under domain restrictions (i.e., the addition of arbitrary unary constraints).*

It is easy to verify from the definition that the broken-triangle property is conservative. This has two important benefits. First, the broken-triangle property is invariant under arc consistency operations: if a binary CSP instance satisfies the broken-triangle property, then so does its arc consistency closure. Second, if the broken-triangle property is satisfied on all triples of variables i, j, k belonging to some subset of variables W, then the CSP instance which results when all of the variables not in W have been assigned will satisfy the broken-triangle property, and hence be efficiently solvable.

3 TRACTABILITY OF BTP INSTANCES

In this section we show that if a CSP instance has the broken-triangle property with respect to some fixed variable ordering, then finding a solution is tractable. Moreover, the problem of finding a suitable ordering if it exists is also tractable.

For a binary CSP instance with n variables, let $d = \max\{|D_1|, \ldots, |D_n|\}$ and let q be the number of constraints.

Definition 6 *An assignment of values $(u_1, \ldots, u_k)$ to the first k variables of a binary CSP instance is called* **consistent** *if $u_i \in D_i$ whenever $1 \leq i \leq k$, and $(u_i, u_j) \in R_{ij}$ whenever $1 \leq i < j \leq k$.*

Theorem 7 *For any binary CSP instance which satisfies the BTP with respect to some known variable ordering $<$, it is possible to find a solution in $O(d^2 q)$ time (or determine that no solution exists).*

Proof: By the discussion above, if an instance has the BTP with respect to $<$, then establishing arc consistency preserves the BTP. Furthermore, it is known that arc consistency can be established in $O(d^2 q)$ time [1]. If this results in an empty domain, then the instance has no solutions. Therefore, we assume in the following that the CSP instance is arc consistent and has non-empty domains.

We can assign some value $u_1 \in D_1$ to the first variable, since $D_1 \neq \emptyset$. To prove the result it is sufficient to show, for all $k = 2, \ldots, n$, that any consistent assignment $(u_1, \ldots, u_{k-1})$ for the first $k-1$ variables can be extended to a consistent assignment $(u_1, \ldots, u_k)$ for the first k variables. The case $k = 2$ follows from arc consistency.

By Lemma 3, if $i < j < k$ then either $R_{ik}(u_i) \subseteq R_{jk}(u_j)$ or $R_{jk}(u_j) \subseteq R_{ik}(u_i)$. Thus the set $\{R_{ik}(u_i) \mid i < k\}$ is totally ordered by subset inclusion, and hence has a minimal element

$$R_{i_0 k}(u_{i_0}) \;=\; \bigcap_{i<k} R_{ik}(u_i) \qquad (2)$$

for some $i_0 < k$. Since the instance is arc consistent, $R_{i_0 k}(u_{i_0}) \neq \emptyset$. By the definition of $R_{ik}(u_i)$, it follows that $(u_1, \ldots, u_k)$ is a consistent assignment for the first k variables, for any choice of $u_k \in R_{i_0 k}(u_{i_0})$.

The time taken to calculate the intersections in (2) is at most $O(d^2 q)$ overall, since each pair of values must be checked against each relevant constraint. ∎

Theorem 8 *The problem of finding a variable ordering for a binary CSP instance such that it satisfies the broken-triangle property with respect to that ordering (or determining that no such ordering exists) is solvable in polynomial time.*

Proof: Given a CSP instance P, we define a new CSP instance P' that has a solution precisely when there exists a suitable variable ordering for P.

To construct P', let $O_1, \ldots, O_n$ be variables taking values in $\{1, \ldots, n\}$ representing positions in the ordering. We impose the ternary constraint

$$O_k \;<\; \max\{O_i, O_j\} \qquad (3)$$

for all triples of variables $i < j < k$ in P such that the broken-triangle property fails to hold for some $u \in D_i$, $v \in D_j$, and $a, b \in D_k$. The instance P' then has a solution precisely if there is an ordering of the variables $1, \ldots, n$ of P which satisfies the broken-triangle property. Note that if the solution obtained represents a partial order (for instance, if O_i and O_j are assigned the same value for some $i \neq j$), then it can be extended to a total order which still satisfies all the constraints by using a linear time topological sort.

For each triple of variables in P, the construction of the corresponding constraints in P' requires $O(d^4)$ steps to check which constraints to add. There are $O(n^3)$ such triples, so constructing instance P' takes $O(n^3 d^4)$ steps, which is polynomial in the size of P.

The constraints in P' are all of the form (3), and such constraints are *max-closed* [14] (if $p_1 < \max\{q_1, r_1\}$ and $p_2 < \max\{q_2, r_2\}$ then $\max(p_1, p_2) \;<\; \max\{\max(q_1, q_2), \max(r_1, r_2)\}$). Max-closed constraints are a tractable constraint language [14]: any CSP instance with max-closed constraints can be solved by establishing generalized arc consistency [15] and then choosing the maximum element which remains in each variable domain. Since the size of P' is polynomial in the size of P, it follows that the instance P' can be solved in time polynomial in the size of P. ∎

Because the BTP is conservative, any pre-processing operations which only perform domain reductions, such as arc consistency, path-inverse consistency [11], or neighbourhood substitution [10, 6], can be applied before looking for a variable ordering for which the broken-triangle property is satisfied; these reduction operations cannot destroy the broken-triangle property, but they can make it more likely to hold (and easier to check).

4 RELATED CLASSES

In this section we will show that the broken-triangle property generalizes two other known tractable classes: one based on language restrictions and one based on structural restrictions.

Throughout this section we suppose that the values in the variable domains are totally ordered.

Definition 9 *A binary relation R_{ij} is* right monotone *if* $\forall b, c \in D_j$,

$$(a, b) \in R_{ij} \;\wedge\; b < c \;\Rightarrow\; (a, c) \in R_{ij}.$$

A commonly-used right monotone constraint is the inequality constraint: $X_i \leq X_j$. The complete relation is also right monotone.

Lemma 10 *If the relations R_{ik}, R_{jk} are both right monotone, then the broken triangle property is satisfied on the triple of variables $i < j < k$, whatever the relation R_{ij}.*

Proof: Suppose that R_{ik}, R_{jk} are both right monotone and that $(u, v) \in R_{ij}$, $(u, a) \in R_{ik}$ and $(v, b) \in R_{jk}$. If $a < b$, then $(u, b) \in R_{ik}$ (since R_{ik} is right monotone); if $a > b$, then $(v, a) \in R_{jk}$ (since R_{jk} is right monotone). ∎

Definition 11 *Consider a binary CSP instance P. For a given variable ordering $<$, denote by $parents_<(k)$ the set of variables $i < k$ such that R_{ik} is proper.*

Definition 12 *A binary CSP instance is* **renamable right monotone** *with respect to a variable ordering $<$ if, for each $k \in \{2, \ldots, n\}$, there is an ordering of D_k, such that R_{ik} is right monotone for every $i \in parents_<(k)$.*

Lemma 13 *If a binary CSP instance is renamable right monotone with respect to a variable ordering $<$, then it satisfies the broken-triangle property with respect to $<$.*

Proof: Suppose the CSP instance is renamable right monotone with respect to variable ordering $<$, and let k be any variable. Since the instance is renamable right monotone with respect to $<$, there is an ordering of D_k such that whenever $i \in parents_<(k)$ then R_{ik} is right monotone. Now suppose $i < j < k$ are variables in this ordering. Then each of R_{ik} and R_{jk} is either the complete relation (and hence right monotone), or right monotone in its own right. By Lemma 10, the broken triangle property is satisfied for i, j, k. Since the choice of k was arbitrary, it follows that the instance satisfies the BTP. ∎

Lemma 14 *If a CSP instance has a tree structure, then it satisfies the broken-triangle property with respect to any variable ordering in which each node occurs before its children.*

Proof: If a CSP instance has tree structure, then any variable ordering $<$ from any designated root to the leaves is such that $|parents_<(k)| \leq 1$ for every variable k. Hence, by Lemma 4, it satisfies the BTP with respect to that ordering. ∎

Let TREE be the constraint satisfaction problem consisting of all instances that have tree structure, RRM be the CSP consisting of all instances that are renamable right monotone w.r.t. some variable ordering, and BTP be the CSP consisting of all instances which have the broken-triangle property w.r.t. some variable ordering. Note that the class RRM contains instances of arbitrary tree-width, for instance some CSPs where the constraint structure is a grid.

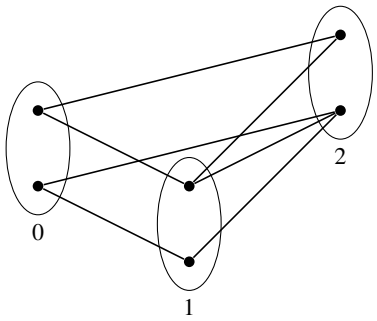

Figure 2. An instance in BTP that is not in RRM or TREE.

Theorem 15 *TREE $\subsetneq$ BTP and RRM $\subsetneq$ BTP.*

Proof: The inclusions follow from Lemma 14 and Lemma 13; the instance shown in Figure 2 establishes the strict separations. ∎

5 ALTERNATIVE CHARACTERIZATION

In this section we consider properties which are both conservative and preserved by taking subproblems. We show that the broken-triangle property is the *only* such property which ensures that the following desirable behaviour can be guaranteed simply by achieving a certain level of arc-consistency:

Definition 16 *A CSP instance is* **universally backtrack-free** *with respect to an ordering $<$ of its n variables if $\forall k \in \{2, \ldots, n\}$, any consistent assignment for the first $k-1$ variables can be extended to a consistent assignment for the first k variables.*

Definition 17 *Given a CSP instance I on variables $1, \ldots, n$, the* **subproblem** *$I(\{i_1, \ldots, i_m\})$, where $1 \leq i_1 < i_2 < \ldots < i_m \leq n$, is the m-variable CSP instance with domains $D_{i_1}, \ldots, D_{i_m}$ and exactly those constraints of I whose scopes are subsets of $\{i_1, \ldots, i_m\}$.*

Definition 18 *A set Σ of CSP instances is* **inclusion-closed** *if $\forall I \in \Sigma$, all subproblems $I(M)$ on subsets M of the variables of I also belong to Σ.*

Definition 19 *A binary CSP instance is* **directional arc-consistent** *with respect to a variable ordering $<$, if for all pairs of variables $i < j$, $\forall a \in D_i$, $\exists b \in D_j$ such that $(a, b) \in R_{ij}$.*

Proposition 20 *A conservative inclusion-closed set Σ of CSP instances is such that the directional arc-consistency closure DAC(I) of every $I \in \Sigma$ with respect to a variable ordering $<$ is universally backtrack-free with respect to $<$ if and only if $\forall I \in \Sigma$, DAC(I) satisfies the broken-triangle property with respect to $<$.*

Proof: The argument used in the proof of Theorem 7 shows that if any binary CSP instance satisfies the broken-triangle property then its directional arc-consistency closure is universally backtrack-free.

To prove the converse, suppose that Σ is a conservative inclusion-closed set of CSP instances and consider any $I \in \Sigma$. Since Σ is conservative, DAC(I) also belongs to Σ, since it is obtained from I by a sequence of domain reductions. In the following, we let D_i denote the domain of variable i in DAC(I). Consider three variables $i < j < k$ and four domain values $u \in D_i$, $v \in D_j$, $a, b \in D_k$ such that $(u, v) \in R_{ij}$, $(u, a) \in R_{ik}$ and $(v, b) \in R_{jk}$. Denote by I' the subproblem of DAC(I) on variables i, j, k and with reduced domain $\{a, b\}$ for variable k. Establishing directional arc consistency in I' may reduce the domains of variables i and j, but cannot delete v from the domain of variable j (since it has a support, namely b, at k) nor can it delete u from the domain of variable i (since it has supports at variables j and k). If DAC(I') is universally backtrack-free, then the consistent assignment (u, v) for the variables (i, j) can be extended to a consistent assignment for (i, j, k), which must be either (u, v, a) or (u, v, b). This corresponds exactly to the definition of the broken-triangle property, and so DAC(I) satisfies the BTP. ∎

6 GENERALIZING THE BTP

In this section we show that a weaker form of the broken-triangle property also implies backtrack-free search. This leads to a larger, but non-conservative, tractable class of CSP instances. Throughout this section, we assume that domains are totally ordered.

Definition 21 *A binary CSP instance is* **min-of-max extendable** *with respect to the variable ordering $<$, if for all triples of variables i, j, k such that $i < j < k$, if $(u, v) \in R_{ij}$, then (u, v, c) is a consistent assignment for (i, j, k), where*

$$c \;=\; \min(\max(R_{ik}(u)), \max(R_{jk}(v)))$$

The symmetrically equivalent property max-of-min extendability is defined similarly, with $c = \max(\min(R_{ik}(u)), \min(R_{jk}(v)))$.

Lemma 22 *A binary CSP instance satisfies the broken-triangle property w.r.t. a variable ordering $<$ if and only if it is min-of-max extendable w.r.t. $<$ for all possible domain orderings.*

Proof: Suppose that a CSP instance satisfies the broken-triangle property with respect to $<$, and consider an arbitrary ordering of each of the domains. To prove min-of-max extendability, it suffices to apply the broken-triangle property to $a = \max(R_{ik}(u))$ and $b = \max(R_{jk}(v))$. Since a and b are maximal, it must be $(u, v, \min(a, b))$ which is the consistent extension of (u, v).

To prove the converse, suppose that a CSP instance is min-of-max extendable for all possible domain orderings. For any $a, b \in D_k$, consider an ordering of D_k for which a, b are the two maximal elements. The broken-triangle property then follows from the definition of min-of-max extendability. ∎

Theorem 23 *If a binary CSP instance is min-of-max extendable w.r.t. some known variable ordering $<$ and some (possibly unknown) domain orderings, and is also directional arc-consistent with respect to $<$, then it is universally backtrack-free w.r.t. $<$, and hence can be solved in polynomial time.*

Proof: Suppose that $(u_1, \ldots, u_{k-1})$ is a consistent assignment for the variables $(1, \ldots, k-1)$. By directional arc consistency, $\forall i < k$, $R_{ik}(u_i) \neq \emptyset$. This means that

$$c \;=\; \min\{\max(R_{ik}(u_i)) : 1 \leq i \leq k-1\}$$

is well-defined. Let $j \in \{1, \ldots, k-1\}$ be such that $c = \max(R_{jk}(u_j))$. Let i be any variable in $\{1, \ldots, k-1\} - \{j\}$. Applying the definition of min-of-max extendability to variables i, j, k allows us to deduce that $(u_i, c) \in R_{ik}$. It follows that $\exists u_k \in D_k$

(namely $u_k = c$) such that $(u_1, \ldots, u_k)$ is a consistent assignment for the variables $(1, \ldots, k)$.

Note that we used the ordering of domain D_k only to prove the existence of a consistent extension $(u_1, \ldots, u_k)$ of $(u_1, \ldots, u_{k-1})$. A backtrack-free search algorithm need not necessarily choose $u_k = c$ and hence does not need to know the domain orderings. ∎

Theorem 24 *The problem of finding a variable ordering for a binary CSP instance with ordered domains such that it is min-of-max extendable w.r.t. that ordering (or determining that no such ordering exists) is solvable in polynomial time.*

Proof: The requirements for the ordering are a subset of the requirements for establishing the broken triangle property. Hence the result can be proved exactly as in the proof of Theorem 8. ∎

We can use Theorem 24 in the following way: given a CSP instance with ordered domains, compute its arc consistency closure, and then test (in polynomial time) whether this reduced instance is min-of-max extendable for some ordering of its variables. If we find such an ordering, then the instance can be solved in polynomial-time, by Theorem 23.

However, this approach is not guaranteed to find all possible useful variable orderings achieving min-of-max extendability. Since min-of-max extendability is not a conservative property, it may be that, for some variable orderings, the directional arc-consistency closure is min-of-max extendable but the full arc-consistency closure is not (or vice versa). In fact we conjecture that, for a given binary CSP instance with fixed domain orderings, determining whether there exists some variable ordering such that the directional arc-consistency closure is min-of-max extendable with respect to that ordering is NP-complete. We also conjecture that determining whether a CSP instance is min-of-max extendable for some unknown domain orderings, even for a fixed variable ordering, is NP-complete.

Finally, we show that min-of-max extendability is a generalization of a previously-identified hybrid tractable class based on row-convex constraints [18].

Definition 25 *A CSP instance is* **row-convex** *(w.r.t. a fixed variable ordering and fixed domain orderings) if for all pairs of variables $i < j$, $\forall u \in D_i$, $R_{ij}(u)$ is the interval $[a, b]$ for some $a, b \in D_j$.*

It is known that a directional path-consistent row-convex binary CSP instance is universally backtrack-free and hence tractable [18]. (However, it should be noted that establishing directional path consistency may destroy row-convexity.) Our interest in this hybrid tractable class is simply to demonstrate that it is a special case of min-of-max extendability.

Proposition 26 *If a binary CSP instance is directional path-consistent and row-convex, then it is min-of-max extendable (and also max-of-min extendable).*

Proof: Consider the triple of variables $i < j < k$ and suppose that $(u, v) \in R_{ij}$. By directional path consistency, $\exists c \in D_k$ such that $(u, c) \in R_{ik}$ and $(v, c) \in R_{jk}$. By row-convexity, $R_{ik}(u)$ and $R_{jk}(v)$ are intervals in the ordered domain D_k. The existence of c means that these intervals overlap. Both end-points of this overlap provide extensions of (u, v) to a consistent assignment for the variables (i, j, k). One end-point is given by $\min(\max(R_{ik}(u)), \max(R_{jk}(v)))$ which ensures min-of-max extendability. (The other ensures max-of-min extendability.) ∎

7 CONCLUSION

We have described new hybrid tractable classes of binary CSP instances which significantly generalize tree-structured problems as well as previously-identified language-based and hybrid tractable classes. The new classes are based on local properties of ordered triples of variables. Moreover, we have shown that the problem of determining a variable ordering for which these properties hold is solvable in polynomial time.

We see this work as a first step towards a complete characterization of all hybrid tractable classes of constraint satisfaction problems.

REFERENCES

[1] C. Bessière and J.-C. Régin, 'Refining the basic constraint propagation algorithm', in *Proc IJCAI'01, Seattle, WA*, pp. 309–315, (2001).

[2] Andrei Bulatov, Peter Jeavons, and Andrei Krokhin, 'Classifying the complexity of constraints using finite algebras', *SIAM Journal on Computing*, **34**(3), 720–742, (2005).

[3] Andrei A. Bulatov, 'Tractable conservative constraint satisfaction problems', in *Proceedings of 18th IEEE Symposium on Logic in Computer Science (LICS 2003), 22-25 June 2003, Ottawa, Canada*, pp. 321–330. IEEE Computer Society, (2003).

[4] Andrei A. Bulatov, 'A dichotomy theorem for constraint satisfaction problems on a 3-element set', *Journal of the ACM*, **53**(1), 66–120, (2006).

[5] David Cohen, Peter Jeavons, and Marc Gyssens, 'A unified theory of structural tractability for constraint satisfaction problems', *Journal of Computer and System Sciences*, **74**(5), 721–743, (2008).

[6] Martin C. Cooper, 'Fundamental properties of neighbourhood substitution in constraint satisfaction problems', *Artificial Intelligence*, **90**(1–2), 1–24, (1997).

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

[8] Rina Dechter, 'Tractable structures for constraint satisfaction problems', in *Handbook of Constraint Programming*, eds., Francesca Rossi, Peter van Beek, and Toby Walsh, 209–244, Elsevier, (2006).

[9] Tomás Feder and Moshe 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).

[10] Eugene C. Freuder, 'Eliminating interchangeable values in constraint satisfaction problems', in *Proc. AAAI-91, Anaheim, CA*, pp. 227–233, (1991).

[11] Eugene C. Freuder and Charles D. Elfe, 'Neighborhood inverse consistency preprocessing', in *Proc. AAAI/IAAI-96, Portland, OR, Vol. 1*, pp. 202–208, (1996).

[12] Martin Grohe, 'The structure of tractable constraint satisfaction problems', in *Proceedings of the 31st Symposium on Mathematical Foundations of Computer Science*, volume 4162 of *Lecture Notes in Computer Science*, pp. 58–72. Springer-Verlag, (2006).

[13] P.G. Jeavons, 'On the algebraic structure of combinatorial problems', *Theoretical Computer Science*, **200**, 185–204, (1998).

[14] P.G. Jeavons and M.C. Cooper, 'Tractable constraints on ordered domains', *Artificial Intelligence*, **79**(2), 327–339, (1995).

[15] R. Mohr and G. Masini, 'Good old discrete relaxation', in *Proceedings 8th European Conference on Artificial Intelligence —ECAI'88*, ed., Y. Kodratoff, pp. 651–656. Pitman, (1988).

[16] J.K. Pearson and P.G. Jeavons, 'A survey of tractable constraint satisfaction problems', Technical Report CSD-TR-97-15, Royal Holloway, University of London, (July 1997).

[17] T.J. Schaefer, 'The complexity of satisfiability problems', in *Proceedings 10th ACM Symposium on Theory of Computing, STOC'78*, pp. 216–226, (1978).

[18] Peter van Beek and Rina Dechter, 'On the minimality and decomposability of row-convex constraint networks', *Journal of the ACM*, **42**(3), 543–561, (1995).