Tractable Disjunctive Constraints

David Cohen[1] , Peter Jeavons[1], and Manolis Koubarakis[2]

[1] Department of Computer Science, Royal Holloway, University of London, UK
e-mail: p.jeavons@dcs.rhbnc.ac.uk
[2] Department of Computation, UMIST, Manchester, UK e-mail:
manolis@sna.co.umist.ac.uk

Abstract. Many combinatorial search problems can be expressed as 'constraint satisfaction problems', and this class of problems is known to be NP-complete in general. In this paper we investigate 'disjunctive constraints', that is, constraints which have the form of the disjunction of two constraints of specified types. We show that when the constraint types involved in the disjunction have a certain property, which we call 'independence', and when a certain restricted class of problems is tractable, then the class of all problems involving these disjunctive constraints is tractable. We give examples to show that many known examples of tractable constraint classes arise in this way, and derive new tractable classes which have not previously been identified.

Keywords: Constraint satisfaction problem, complexity, NP-completeness

1 Introduction

The constraint satisfaction problem provides a framework in which it is possible to express, in a natural way, many combinatorial problems encountered in artificial intelligence and elsewhere. The aim in a constraint satisfaction problem is to find an assignment of values to a given set of variables subject to constraints on the values which can be assigned simultaneously to certain specified subsets of variables.

The constraint satisfaction problem is known to be an NP-complete problem in general [19]. However, by imposing restrictions on the constraint interconnections [4, 5, 7, 20, 21], or on the form of the constraints [1, 20, 2, 9, 12, 25, 26], it is possible to obtain restricted versions of the problem which are tractable.

Now that a number of constraint types have been identified which give rise to tractable subproblems of the constraint satisfaction problem, it is of considerable interest to investigate how tractable constraint types may be combined, to yield more general problems which are still tractable.

In this paper we investigate 'disjunctive constraints', that is, constraints which have the form of the disjunction of two constraints of specified types. We show that when the constraint types involved in the disjunction have a certain property, which we call 'independence', and when a certain very restricted subproblem is tractable, then the class of all problems involving these disjunctive constraints is also tractable. This allows new tractable constraint classes to be constructed from existing classes, and so extends the range of known tractable constraint classes.

We give examples to show that many known examples of tractable disjunctive constraints over both finite and infinite domains arise in this way. In particular, we demonstrate that two of the six tractable subproblems of the 'Generalised Satisfiability' problem identified by Schaefer in [23] can be obtained by combining simpler tractable subproblems in this standard way. Furthermore, we show that similar results hold for the 'max-closed' constraints first identified in [2], the ORD-Horn constraints over temporal intervals described in [22], and the disjunctive linear constraints over the real numbers described in [15]. Finally, we describe a new tractable class of constraints over the integers which is derived from the same theorem.

The paper is organised as follows. In Section 2 we give the basic definitions for the constraint satisfaction problem, and define the notion of a tractable set of relations. In Section 3 we introduce the notion of 'independence' of one set of relations with respect to another, and show that this condition, together with the tractability of a certain restricted subproblem, is sufficient to ensure tractability of disjunctive constraints involving two such independent sets. In Section 4 we give examples to illustrate how this property may be used to establish the tractability of a wide variety of tractable constraint classes.

2 Definitions

2.1 The constraint satisfaction problem

Notation 1 *For any set D, and any natural number n, we denote the set of all n-tuples of elements of D by D^n. For any tuple $t \in D^n$, and any i in the range 1 to n, we denote the value in the ith coordinate position of t by $t[i]$. The tuple t may then be written in the form $\langle t[1], t[2], \ldots, t[n] \rangle$.*

A subset of D^n is called an *'n-ary relation'* over D.

We now define the 'constraint satisfaction problem' which has been widely studied in the Artificial Intelligence community [19, 20, 16]

Definition 2. An instance of a *constraint satisfaction problem* consists of:

- A finite set of variables, V;
- A domain of values, D;
- A finite set of constraints $C = \{c_1, c_2, \ldots, c_q\}$.
  Each constraint c_i is a pair (S_i, R_i), where S_i is a list of variables of length m_i, called the 'constraint scope', and R_i is an m_i-ary relation over D, called the 'constraint relation'.

The tuples of a constraint relation indicate the allowed combinations of simultaneous values for the variables in the constraint scope. The length of the tuples in the constraint relation will be called the 'arity' of the constraint. In particular, unary constraints specify the allowed values for a single variable, and binary constraints specify the allowed combinations of values for a pair of variables.

To determine the computational complexity of a constraint satisfaction problem we need to specify how each problem instance is encoded in a finite string. In order to simplify this representation, we shall impose the condition that every variable occurs in the scope of some constraint, and hence the set of variables does not need to be specified explicitly, but is given by the union of the constraint scopes. Furthermore, we shall assume that the domain of values is specified implicitly in the definition of the constraint relations. Using these assumptions, we shall specify a constraint satisfaction problem instance by specifying the corresponding set of constraints.

In this specification, each constraint relation R_i is represented by a finite string of symbols $\rho(R_i)$, and we assume that this representation is chosen so that the complexity of determining whether any given tuple is an element of R_i is bounded by a polynomial function of the length of $\rho(R_i)$. A *finite* relation can be represented simply by giving an explicit list of all the tuples it contains, but for infinite relations we need to use a suitable specification language, such as logical formulas, or linear equations.

A *solution* to a constraint satisfaction problem instance is a function from the variables to the domain such that the image of each constraint scope is an element of the corresponding constraint relation. Since we are representing instances by specifying the set of constraints of that instance, we shall talk about solutions to a set of constraints. The set of all solutions to the set of constraints C will be denoted $\mathrm{Sol}(C)$.

Deciding whether or not a given set of constraints has a solution is NP-complete in general [19]. In this paper we shall consider how restricting the allowed constraint relations to some fixed subset of all the possible relations affects the complexity of this decision problem. We therefore make the following definition.

Definition 3. For any set of relations, Γ, $\mathbf{CSP}(\Gamma)$ is defined to be the decision problem with

INSTANCE: A finite set of constraints C, in which all constraint relations are elements of Γ.
QUESTION: Does C have a solution?

If there is some algorithm which solves every instance in $\mathrm{CSP}(\Gamma)$ in polynomial time, then we shall say that $\mathrm{CSP}(\Gamma)$ is 'tractable', and refer to Γ as a tractable set of relations.

Example 1. The binary inequality relation over a set D is defined as follows.

$$\neq_D = \{\langle d_1, d_2 \rangle \in D^2 \mid d_1 \neq d_2\}$$

$\mathrm{CSP}(\{\neq_D\})$ corresponds precisely to the Graph Colorability problem [6]. This problem is well-known to be tractable when $|D| \leq 2$ and NP-complete when $|D| \geq 3$.

In order to construct new tractable sets of relations, we will make use of the following operation to combine two sets of relations.

Definition 4. For any set of relations Γ, let Γ_i denote the subset of Γ containing all relations in Γ with arity i, together with the empty relation.

For any sets of relations Γ and Δ, define the set of relations $\Gamma \overset{\times}{\vee} \Delta$ as follows:

$$\Gamma \overset{\times}{\vee} \Delta = \{R \cup R' \mid \exists i, R \in \Gamma_i, R' \in \Delta_i\}$$

Note that the set of relations $\Gamma \overset{\times}{\vee} \Delta$ contains all the members of Γ, all the members of Δ, and the union of each possible *pair* of relations from Γ and Δ with the same arity. This set is, in general, much larger than the set $\Gamma \cup \Delta$, and hence able to express many more constraints. The next example shows that when tractable sets of relations are combined using the disjunction operation defined in Definition 4 the resulting set of relations may or may not be tractable.

Example 2. Let Λ be the the set of all Boolean relations which can be specified by a formula of propositional logic consisting of a single literal (where a literal is either a variable or a negated variable).

The subset of relations Λ_1 contains the following two relations (together with the empty relation):

- $\{\langle T\rangle\}$, corresponding to the literal p_1;
- $\{\langle F\rangle\}$, corresponding to the literal $\neg p_1$.

To specify relations of higher arity with a single literal, we associate the literal with one coordinate position in the relation, and allow arbitrary values in the other positions. This means, for example, that the subset Λ_2 contains the following four relations (together with the empty relation):

- $\{\langle T,T\rangle, \langle T,F\rangle\}$, corresponding to the literal p_1;
- $\{\langle F,T\rangle, \langle F,F\rangle\}$, corresponding to the literal $\neg p_1$;
- $\{\langle T,T\rangle, \langle F,T\rangle\}$, corresponding to the literal p_2;
- $\{\langle T,F\rangle, \langle F,F\rangle\}$, corresponding to the literal $\neg p_2$.

The set of relations Λ is clearly tractable, as it is straightforward to verify in linear time whether a collection of literals has a simultaneous solution.

Now consider the set of relations $\Lambda^2 = \Lambda \overset{\times}{\vee} \Lambda$. The class of problems CSP(Λ^2) corresponds to the 2-Satisfiability problem, which is well-known to be tractable [6].

Finally, consider the set of relations $\Lambda^3 = \Lambda \overset{\times}{\vee} \Lambda^2$. The problem CSP($\Lambda^3$) corresponds to the 3-Satisfiability problem, which is well-known to be NP-complete [6].

When we use the set of relations $\Gamma \overset{\times}{\vee} \Delta$ as constraint relations, we shall assume that the constraint relations in any given problem instance are represented by strings of symbols of the form '$\rho(R) \vee \rho(R')$', for some R in Γ and some R' in Δ. With this convention, we can define two decomposition operations, Π_1 and Π_2 on constraints in $\mathrm{CSP}(\Gamma \overset{\times}{\vee} \Delta)$ which pick out the two components in the disjunction. These operations are defined as follows: for any constraint $c = (S_c, R_c)$, where R_c is represented by the string '$\rho(R) \vee \rho(R')$', we set $\Pi_1(c) = (S_c, R)$ and $\Pi_2(c) = (S_c, R')$. Note that the operations Π_1 and Π_2 can be carried out in linear time.

3 Tractable Disjunctive Constraints

In this section we shall identify certain conditions on sets of relations Γ and Δ which are sufficient to ensure that $\Gamma \overset{\times}{\vee} \Delta$ is tractable. To describe these conditions we will make use of the following definition:

Definition 5. For any sets of relations Γ and Δ, define $\mathrm{CSP}_{\Delta\leq 1}(\Gamma \cup \Delta)$ to be the subproblem of $\mathrm{CSP}(\Gamma \cup \Delta)$ consisting of all instances containing at most one constraint whose constraint relation is a member of Δ.

Using this definition, we now define what it means for one set of relations to be 'independent' with respect to another.

Definition 6. For any sets of relations Γ and Δ, we say that Δ is *independent with respect to* Γ if for any set of constraints C in $\mathrm{CSP}(\Gamma \cup \Delta)$, C has a solution whenever every subset C' of C which belongs to $\mathrm{CSP}_{\Delta\leq 1}(\Gamma \cup \Delta)$ has a solution.

The intuitive meaning of this definition is that the satisfiability of any set of constraints from Δ may be determined by considering those constraints one at a time, even in the presence of arbitrary additional constraints from Γ. In the examples below we shall demonstrate that several important constraint types have this independence property.

A more restrictive notion of independence was described in [18]. This earlier notion of independence was defined for a single set of relations, and has been used to establish the tractability of a number of constraint types. However, we will show below that the more general notion of independence introduced here can be used to prove the tractability of a wide variety of *disjunctive* constraint types for which the earlier notion does not hold.

First, we show that this independence property is preserved over arbitrary disjunctions, in the sense defined by the following lemma.

Lemma 7. *For any set of relations Δ, define the set Δ^* as follows:*

$$\Delta^* = \bigcup_{i=1}^{\infty} \Delta^i$$

where

$$\begin{aligned} \Delta^1 &= \Delta \\ \Delta^{i+1} &= \Delta^i \overset{\times}{\vee} \Delta \quad \textit{for} \;\; i = 1, 2, 3, \ldots \end{aligned}$$

If Δ is independent with respect to Γ, then Δ^* is also independent with respect to Γ.

Proof. Assume that Δ is independent with respect to Γ. In order to show that Δ^* is independent of Γ, let C be any set of constraints in $\mathrm{CSP}(\Gamma \cup \Delta^*)$. We need to show that if every subset of C which belongs to $\mathrm{CSP}_{\Delta^* \leq 1}(\Gamma \cup \Delta^*)$ has a solution, then so does C.

Let C' be a maximal subset of C belonging to $\mathrm{CSP}_{\Delta^* \leq 1}(\Gamma \cup \Delta^*)$ and let s be a solution to C'. Since C' is maximal, it contains the set C_Γ, consisting of all the constraints in C whose constraint relations are elements of $\Gamma \setminus \Delta^*$, so s is a solution to C_Γ. If C' also contains a constraint $d = (S_d, R_d)$ whose constraint relation, R_d, is an element of Δ^*, then there must be at least one relation R'_d in Δ such that s is a solution to the constraint $d' = (S_d, R'_d)$, by the definition of Δ^*. Hence, we may replace d with a (possibly) more restrictive constraint d' whose constraint relation is an element of Δ, without losing the solution s.

If we carry out this replacement for each C', then we have a set of constraints in $\mathrm{CSP}(\Gamma \cup \Delta)$ such that each subset belonging to $\mathrm{CSP}_{\Delta \leq 1}(\Gamma \cup \Delta)$ has a solution. Now, by the fact that Δ is independent with respect to Γ, it follows that this modified set of constraints has a solution, and hence the original set of constraints, C, has a solution.

Now consider the following algorithm for a function **Solvable** which determines whether or not a set of constraints C, with relations chosen from $\Gamma \overset{\times}{\vee} \Delta$, has a solution:

Algorithm 8 *A function to determine if a set of constraints has a solution:*

Solvable(C: SET OF CONSTRAINTS IN $\mathrm{CSP}(\Gamma \overset{\times}{\vee} \Delta)$)
  $F := \emptyset$
  REPEAT
    $X := \{c \in C \mid \mathrm{Sol}(F \cup \Pi_2(c)) = \emptyset\}$
    IF $X = \emptyset$ THEN
      RETURN TRUE
    ELSE
      $C := C \setminus X$
      $F := F \cup \{\Pi_1(x) \mid x \in X\}$
    ENDIF
  UNTIL $\mathrm{Sol}(F) = \emptyset$
  RETURN FALSE

The next result shows that Algorithm 8 correctly determines whether or not a set of constraints with relations chosen from $\Gamma \overset{\times}{\vee} \Delta$ has a solution in those cases when Δ is independent with respect to Γ.

Theorem 9. *If C is an element of CSP($\Gamma \overset{\times}{\vee} \Delta$) and Δ is independent with respect to Γ, then Algorithm 8 correctly determines whether or not C has a solution.*

Proof. Algorithm 8 clearly terminates, because C is finite.

Assume that C is an element of CSP($\Gamma \overset{\times}{\vee} \Delta$), for some Γ and Δ such that Δ is independent with respect to Γ. We first prove by induction that after every assignment to F, all of the constraints in F must be satisfied in order to satisfy the original set of constraints, C.

This is vacuously true after the first assignment to F, because F is then equal to the empty set.

At each subsequent assignment, F is augmented with the constraints obtained by applying Π_1 to the constraints in X. Now, the elements of X are constraints c of C such that $\Pi_2(c)$ is incompatible with F. Hence the only way such a c can be satisfied together with the current constraints in F is to satisfy the other disjunct of the constraint relation of c, or in other words, the constraint given by $\Pi_1(c)$. Hence, by the inductive hypothesis, each constraint added to F must be satisfied in order to satisfy the original set of constraints C, and the result follows, by induction.

This result establishes that when **Solvable**(C) returns FALSE, C has no solutions.

Conversely, when **Solvable**(C) returns TRUE, then we know that X is empty. This implies that for each constraint c in C either $\Pi_1(c)$ belongs to F, or else $\Pi_2(c)$ is compatible with the constraints in F. Now, using the fact that Δ is independent with respect to Γ, we conclude that $F \cup \{\Pi_2(c) \mid c \in C,\ \Pi_1(c) \notin F\}$ has a solution, and hence C has a solution.

By analysing the complexity of Algorithm 8 we now establish the following result:

Theorem 10. *For any sets of relations Γ and Δ, if $CSP_{\Delta \leq 1}(\Gamma \cup \Delta)$ is tractable, and Δ is independent with respect to Γ, then CSP($\Gamma \overset{\times}{\vee} \Delta$) is tractable.*

Proof. We can bound the time complexity of Algorithm 8 as follows.

First note that $|F|$ increases on each iteration of the repeat loop, but $|F|$ is bounded by $|C|$ since each constraint in F arises from an element of C. Hence there can be at most $|C|$ iterations of this loop.

Now let $l(C)$ be the length of the string specifying C. During each iteration of the loop the algorithm determines whether or not there is a solution to $F \cup \Pi_2(c)$ for each c remaining in C. Since this set of constraints is a member of $\mathrm{CSP}_{\Delta \leq 1}(\Gamma \cup \Delta)$, which is assumed to be tractable, these calculations may each be carried out in polynomial time in the size of their input. Note also that the length of this input is less than or equal to $l(C)$. Hence, the time complexity of each of these calculations is $O(l(C)^k)$, for some constant k.

At the end of each iteration of the loop the algorithm determines whether or not there is a solution to F. Since F is also an element of $\mathrm{CSP}_{\Delta\leq 1}(\Gamma \cup \Delta)$, and the length of the specification of F is less than or equal to $l(C)$, this calculation can also be carried out in $O(l(C)^k)$ time.

Hence, the total time required to complete the algorithm is $O(|C|(|C|+1)l(C)^k)$, which is polynomial in the size of the input.

4 Applications

In this section we will demonstrate that many known tractable sets of relations can be obtained by combining simpler tractable sets of relations using the disjunction operation, as described in the previous section. We will also describe some new tractable sets of relations which have not previously been identified.

Example 3. [**Horn clauses**] One set of disjunctive constraints which are well-known to be tractable is the set of constraints over Boolean variables which ca be expressed using Horn clauses, that is, disjunctions of literals containing at most one positive literal. We will now show that the tractability of Horn clauses is a simple consequence of Theorem 10.

Recall the set of Boolean relations, Λ, defined in Example 2, which may be specified by a propositional formula consisting of a single literal.

Define Λ_+ to be the subset of Λ which may be specified using a single *positive* literal, together with the empty relation, and Λ_- to be the subset of Λ which may be specified using a single *negative* literal, together with the empty relation.

Now set

$$\Lambda_-^1 = \Lambda_-$$

$$\Lambda_-^{i+1} = \Lambda_-^i \overset{\times}{\vee} \Lambda_- \text{ for } i = 1, 2, 3, \ldots$$

and

$$\Lambda_-^* = \bigcup_{i=1}^{\infty} \Lambda_-^i$$

It is clear that Λ_-^* is equal to the set of relations which may be specified by arbitrary finite disjunctions of negative literals.

Finally, it is easily shown that Λ_- is independent with respect to Λ_+, and hence Λ_-^* is independent with respect to Λ_+ by Lemma 7. Furthermore, $\mathrm{CSP}_{\Lambda_-^*\leq 1}(\Lambda_+ \cup \Lambda_-^*)$ is tractable, since each instance is specified by a conjunction of single positive literals together with at most one disjunction of negative literals. Hence, $\Theta = \Lambda_+ \overset{\times}{\vee} \Lambda_-^*$ is tractable, by Theorem 10. But Θ is the set of relations which may be specified by a disjunction of literals containing at most one positive literal, so $\mathrm{CSP}(\Theta)$ corresponds to the HORN-CLAUSE SATISFIABILITY problem [23, 6].

By symmetry, it follows that the set of relations which may be specified by a disjunction of literals containing at most one negative literal is also a tractable set of relations.

Example 4. **[Max-closed constraints]**

The class of constraints over finite domains known as 'max-closed' constraints was introduced in [2] and shown to be tractable. These constraints are defined over arbitrary finite domains which are totally ordered.

In this example we will show that this result may be obtained as a simple consequence of Theorem 10. Furthermore, by using Theorem 10 we are able to generalise the result to infinite domains.

The max-closed constraints were originally defined in terms of an algebraic closure property on the constraint relations [2]. However, it is shown in [2] that they can also be characterised as those constraints where the constraint relation can be specified by means of a conjunction of disjunctions of inequalities of the following form:

$$(x_1 < a_1) \vee (x_2 < a_2) \vee \cdots \vee (x_r < a_r) \vee (x_j > a_j)$$

In this expression the x_i are variables and the a_i are constants.

Now let D be any domain which is totally ordered, Proceeding as in the previous example, we define $\Gamma_<$ to be the set of relations over D which may be specified by a single inequality of the form $x_i < a_i$, and $\Gamma_>$ to be the set of relations over D which may be specified by a single inequality of the form $x_i > a_i$. Then we set

$$\begin{aligned} \Gamma_<^1 &= \Gamma_< \\ \Gamma_<^{i+1} &= \Gamma_<^i \overset{\times}{\vee} \Gamma_< \quad \text{for } i = 1, 2, 3, \ldots \end{aligned}$$

and

$$\Gamma_<^* = \bigcup_{i=1}^{\infty} \Gamma_<^i$$

It is clear that $\Gamma_<^*$ is equal to the set of relations which may be specified by arbitrary finite disjunctions of inequalities of the form $x_i < a_i$.

Finally, it is easily shown that $\Gamma_<$ is independent with respect to $\Gamma_>$, so we know that $\Gamma_<^*$ is also independent of $\Gamma_>$, by Lemma 7. Also, $\mathrm{CSP}_{\Gamma_<^* \leq 1}(\Gamma_> \cup \Gamma_<^*)$ is tractable, since each instance consists of a conjunction of lower bounds for certain variables together with at most one disjunction of upper bounds. Hence, $\Gamma_> \overset{\times}{\vee} \Gamma_<^*$ is a tractable set of relations, by Theorem 10, which establishes that max-closed constraints are tractable, by the result above.

Unlike the arguments used previously to establish that max-closed constraints are tractable [2, 9], all of the techniques used here can still be applied when the domain, D, is infinite.

Example 5. **[Linear Horn constraints]**

The class of constraints over the real numbers known as 'linear Horn' constraints was introduced in [15] (and independently in [10]) and shown to be tractable.

A *linear Horn constraint* is specified by a disjunction of weak linear inequalities and linear disequations where the number of inequalities does not exceed one. The following are examples of linear Horn constraints:

$$\begin{gathered}3x_1 + x_5 - 3x_4 \leq 10,\ \ x_1 + x_3 + x_5 \neq 7,\\ 3x_1 + x_5 - 4x_3 \leq 7 \vee 2x_1 + 3x_2 - 4x_3 \neq 4 \vee x_2 + x_3 + x_5 \neq 7,\\ 4x_1 + x_3 \neq 3 \vee 5x_2 - 3x_5 + x_4 \neq 6\end{gathered}$$

Linear Horn constraints form an important class of linear constraints with explicit connections to temporal reasoning [10]. In particular, the class of linear Horn constraints properly includes the point algebra of [27], the (quantitative) temporal constraints of [13, 14], and the ORD-Horn constraints of [22]. All these classes of temporal constraints can therefore be shown to be tractable using the framework developed here.

Let the domain D be $\Re$ or $\mathcal{Q}$. Define Γ to be the set of relations defined by a single (weak) linear inequality (e.g., $3x_1 + 2x_2 - x_3 \leq 6$). Define Δ to be the set of relations defined by a single linear disequality (e.g., $x_1 + 4x_2 + x_3 \neq 0$).

Note that Δ^* is the set of relations defined by a disjunction of disequalities, and the problem CSP($\Gamma \cup \Delta^*$) corresponds to deciding whether a convex polyhedron *minus* the union of a finite number of hyperplanes is the empty set. It was shown in [17] that the set $\Gamma \cup \Delta^*$ is independent (using their more restrictive notion of independence referred to above), and hence that this problem is tractable.

However, the set of relations specified by linear Horn constraints corresponds to the much larger set $\Gamma \overset{\times}{\vee} \Delta^*$, and this set is *not* independent in the sense defined in [17] (see [15]). In order to establish that this larger set of constraints is tractable we shall use the more general notion of independence introduced in this paper.

Consider any set of constraints C in CSP($\Gamma \cup \Delta$). Let C' be the subset of C which is specified by weak linear inequalities, and let C'' be the subset of C which is specified by linear disequations. It is clear that C is consistent if and only if C' is consistent, and, for each $c \in C''$, the set $C' \cup \{c\}$ is consistent. Hence, Δ is independent with respect to Γ. By Lemma 7 it follows that Δ^* is also independent with respect to Γ.

Now Theorem 9 implies that Algorithm 8 can be used to determine whether CSP($\Gamma \overset{\times}{\vee} \Delta^*$) has a solution. (In fact, Algorithm 8 can be seen as a simplification of the algorithm Consistency which was developed specifically for this problem in [15].)

Finally, to establish tractability, we note that whether a set of inequalities, C', is consistent or not can be decided in polynomial time, using Khachian's linear programming algorithm [11]. Furthermore, for any single disjunction of disequalities, d, we can detect in polynomial time whether $C' \cup \{d\}$ is consistent by simply running Khachian's algorithm to determine whether C' implies each equality in the conjunction of equalities corresponding to the negation of d. Hence, $\mathrm{CSP}_{\Delta^* \leq 1}(\Gamma \cup \Delta^*)$ is tractable, so by Theorem 10 we know that $\mathrm{CSP}(\Gamma \overset{\times}{\vee} \Delta^*)$ is tractable.

Having identified the key property of independence, which underlies all of these tractable classes, we are now in a position to identify new tractable classes simply by searching for appropriate sets of tractable relations which have this property.

Example 6. **[Disjunctive congruences]**

One of the fundamental results of elementary number theory is the Chinese Remainder Theorem [8], which states that it is always possible to solve a collection of simultaneous linear congruences with co-prime moduli.

This result can be used to obtain tractable constraint classes in a number of ways. One of the most straightforward is to define, for each integer a, the set of unary relations $\Gamma(a)$, containing all relations of the form $\{\langle x \rangle \mid x \equiv a \pmod{p}\}$, for some prime $p > |a|$. A problem instance in $\mathrm{CSP}(\Gamma(a))$ may then be specified by a collection of simultaneous congruences. For example, a typical set of constraints in $\mathrm{CSP}(\Gamma(3))$, involving the variables x_1 and x_2, would be:

$$\begin{aligned} x_1 &\equiv 3 \pmod{5}, \\ x_2 &\equiv 3 \pmod{7}, \\ x_2 &\equiv 3 \pmod{11}. \end{aligned}$$

The Chinese remainder theorem implies that any set of constraints in $\mathrm{CSP}(\Gamma(a))$ has a solution, for any a, so each $\Gamma(a)$ is a tractable set of relations.

Furthermore, any set of constraints in $\mathrm{CSP}(\Gamma(a) \cup \Gamma(b))$ will also have a solution, unless it contains a pair of constraints c_1 and c_2 which are incompatible. (The constraints c_1 and c_2 are incompatible if they both constrain the same variable x with respect to the same prime p, but c_1 requires that $x \equiv a \pmod{p}$ and c_2 requires that $x \equiv b \pmod{p}$, and p does not divide $a - b$.) Hence, for any a and b the set $\Gamma(a)$ is independent of the set $\Gamma(b)$, and vice versa.

Finally, the set of problems $\mathrm{CSP}_{\Gamma(b) \leq 1}(\Gamma(a) \cup \Gamma(b))$ is tractable, because we can determine whether or not any given instance has a solution by examining each pair of constraints to see whether they are compatible. If they are all compatible then the instance has a solution, by the Chinese Remainder Theorem, otherwise it does not.

Hence, by Lemma 7 and Theorem 10, we know that $\mathrm{CSP}(\Gamma(a) \overset{\times}{\vee} \Gamma(b)^*)$ is also tractable, for any choice of a and b. This means that there is an efficient way to solve any collection of simultaneous disjunctions of congruences which all have the property that at most one disjunct comes from $\Gamma(a)$ and the remainder from $\Gamma(b)$. For example, when $a = 1$ and $b = 2$ we might have the following collection of congruences:

$$x_1 \equiv 1 \pmod{2} \vee x_1 \equiv 2 \pmod{3} \vee x_2 \equiv 2 \pmod{5},$$

$$x_2 \equiv 2 \pmod 3 \vee x_3 \equiv 2 \pmod 5,$$
$$x_3 \equiv 1 \pmod 2,$$
$$x_2 \equiv 1 \pmod 7 \vee x_1 \equiv 2 \pmod{11} \vee x_3 \equiv 2 \pmod 3$$

5 Conclusion

In this paper we have examined the complexity of disjunctive constraints, and established sufficient conditions to ensure tractability. We have shown that these conditions account for a wide variety of known tractable disjunctive classes, and that they aid the search for new tractable constraint classes.

References

1. Cooper, M.C., Cohen, D.A., Jeavons, P.G., "Characterizing tractable constraints", *Artificial Intelligence 65*, (1994), pp. 347–361.
2. Cooper, M.C., & Jeavons, P.G., "Tractable constraints on ordered domains", *Artificial Intelligence 79*, (1995), pp. 327–339.
3. Dechter, R., & Pearl, J., "Structure identification in relational data", *Artificial Intelligence 58* (1992) pp. 237–270.
4. Dechter, R. & Pearl J. "Network-based heuristics for constraint-satisfaction problems", *Artificial Intelligence 34* (1988), pp. 1–38.
5. Freuder, E.C., "A sufficient condition for backtrack-bounded search", *Journal of the ACM 32* (1985) pp. 755–761.
6. Garey, M.R., & Johnson, D.S., *Computers and intractability: a guide to NP-completeness*, Freeman, San Francisco, California, (1979).
7. Gyssens, M., Jeavons, P., Cohen, D., "Decomposing constraint satisfaction problems using database techniques", *Artificial Intelligence 66*, (1994), pp. 57–89.
8. Jackson, T.H., *Number Theory*, Routledge and Kegan Paul, (1975).
9. Jeavons, P., Cohen D., Gyssens, M., "A unifying framework for tractable constraints", In *Proceedings 1st International Conference on Principles and Practice of Constraint Programming—CP '95* (Cassis, France, September 1995), *Lecture Notes in Computer Science*, 976, Springer-Verlag, Berlin/New York, 1995, pp. 276–291.
10. Jonsson, P. and Bäckström, C., "A Linear Programming Approach to Temporal Reasoning", Proceedings of AAAI-96, (1996).
11. Khachian, L.G., "A polynomial time algorithm for linear programming", *Soviet Math. Dokl. 20*, (1979) pp. 191–194.
12. Kirousis, L., "Fast parallel constraint satisfaction", *Artificial Intelligence 64*, (1993), pp. 147–160.
13. Koubarakis, M., "Dense Time and Temporal Constraints with $\neq$", In *Principles of Knowledge Representation and Reasoning: Proceedings of the Third International Conference (KR'92)*, (Ed. B. Nebel, C. Rich and W. Swartout), Morgan Kaufmann, San Mateo, CA, (1992), pp. 24–35.
14. Koubarakis, M., "From Local to Global Consistency in Temporal Constraint Networks", In *Proceedings 1st International Conference on Principles and Practice of Constraint Programming—CP '95* (Cassis, France, September 1995), *Lecture Notes in Computer Science*, 976, Springer-Verlag, Berlin/New York, (1995), pp. 53–69.

15. Koubarakis, M., "Tractable Disjunctions of Linear Constraints", In *Proceedings of the 2nd International Conference on Principles and Practice of Constraint Programming—CP '96* (Boston, MA, August 1996), *Lecture Notes in Computer Science*, 1118, Springer-Verlag, Berlin/New York, (1996), pp. 297–307.
16. Ladkin, P.B., & Maddux, R.D., "On binary constraint problems", *Journal of the ACM 41* (1994), pp. 435–469.
17. Lassez, J-L., and McAloon, K., "A Canonical Form for Generalized Linear Constraints", Technical Report RC15004 (#67009), IBM Research Division, T.J. Watson Research Center, (1989).
18. Lassez, J-L., and McAloon, K., "A Canonical Form for Generalized Linear Constraints", *Lecture Notes in Computer Science*, 351, Springer-Verlag, Berlin/New York, (1989), pp. 19–27.
19. Mackworth, A.K. "Consistency in networks of relations", *Artificial Intelligence 8* (1977) pp. 99–118.
20. Montanari, U., "Networks of constraints: fundamental properties and applications to picture processing", *Information Sciences 7* (1974), pp. 95–132.
21. Montanari, U., & Rossi, F., "Constraint relaxation may be perfect", *Artificial Intelligence 48* (1991), pp. 143–170.
22. Nebel, B. & Burckert, H-J., "Reasoning about temporal relations: a maximal tractable subclass of Allen's interval algebra", *Journal of the ACM 42*, (1995), pp. 43–66.
23. Schaefer, T.J., "The complexity of satisfiability problems", *Proc 10th ACM Symposium on Theory of Computing (STOC)* , (1978) pp. 216–226.
24. Tsang, E., *Foundations of Constraint Satisfaction*, Academic Press, (1993).
25. van Beek, P., "On the Minimality and Decomposability of Row-Convex Constraint Networks", *Proceedings of the Tenth National Conference on Artificial Intelligence, AAAI-92*, MIT Press, (1992) pp. 447–452.
26. Van Hentenryck, P., Deville, Y., Teng, C-M., "A generic arc-consistency algorithm and its specializations", *Artificial Intelligence 57* (1992), pp. 291–321.
27. Vilain, M., Kautz, H., van Beek, P., "Constraint Propagation Algorithms for Temporal Reasoning: A Revised Report", In *Readings in Qualitative Reasoning about Physical Systems*, Ed. Weld, D.S. and de Kleer, J., Morgan Kaufmann, (1989), pp. 373–381.