On Guaranteeing Polynomially Bounded Search Tree Size

David A. Cohen[1], Martin C. Cooper[2], Martin J. Green,[1], and Dániel Marx[3]

[1] Dept. of Computer Science, Royal Holloway, University of London, UK
[2] IRIT, University of Toulouse III, 31062 Toulouse, France
[3] Institut für Informatik, Humboldt-Universität zu Berlin, Germany

Abstract. A constraint network consists of a set of variables which must take values in some domain, where the allowed assignments are limited by a set of constraints. Each constraint limits the simultaneous assignment of values to some subset of the variables called its scope. The nature of the restriction given by a particular constraint is called its relation. The structure of a constraint network is the hypergraph over the set of variables whose edges are the constraint scopes. The language of a constraint network is the set of relations over the domain used in its constraints.

Much work has been done on describing tractable classes of constraint networks. Most of the known tractable examples are described by either restricting the structure of the networks, or their language. Indeed, for both structural or language restrictions very strong dichotomy results have been proven and in both cases it is likely that all practical examples have already been discovered.

As such it is timely to consider tractability which cannot be described by language or structural restrictions. This is the focus of the work here.

In this paper we investigate a novel reason for tractability: having at least one variable ordering for which the number of partial solutions to the first n variables is bounded by a polynomial in n.

We show that the presence of sufficient functional constraints can guarantee this property and we investigate the complexity of finding good variable orderings based on different notions of functionality.

What is more we identify a completely novel reason for tractability based on so called Turan sets.

keywords: constraint satisfaction, satisfiability, hybrid tractability, functional constraints, Turan tractability, variable ordering.

1 Introduction

A *constraint network* consists of a collection of *variables*, each of which must take its value from a specified *domain*. Some subsets of these variables have a further limitation, called a *constraint*, on the values they may simultaneously take. Thus a constraint has two components: a list of variables called its *scope*, and a set of allowed valuations that this list may be assigned, called its *relation* [10].

The set of relations occurring in a particular constraint network is often called the *language* of the network. The scopes of a constraint network, abstracted as sets of variables, are called the *structure* of the network.

It is natural to define tractable classes of constraint networks by restricting the language or the structure. It is conjectured that there is a dichotomy for all constraint languages: they are either tractable or NP-hard [7]. This dichotomy has been made explicit by Bulatov [2]. Since much work has been done in this area we have a growing base of evidence for this conjecture.

On the other hand, the work on structural tractability is even further advanced. Grohe [8] showed that a set of structures is tractable if and only if the tree-width of their cores is bounded.

In this paper we extend the classical theory of constraint network tractability beyond the artificial distinction between language and structure. Such *hybrid* tractability is just beginning to be systematically studied [3, 11, 13].

We will exhibit two novel polytime-testable hybrid properties of constraint networks which guarantee a polynomial-size search tree as well as a polynomial number of solutions. Classes defined by such properties are clearly tractable. Indeed, in such cases, polytime solvability is preserved even after the addition of any number of arbitrary constraints and/or the addition of any polytime objective function, since it suffices to test the extra constraints and/or evaluate the objective function for each of the polynomial number of solutions.

We say that a constraint is *functional* [6] on one of its variables if the value of this variable is uniquely determined by an assignment to the rest of the variables of the scope. Examples include functional dependencies in databases, or mathematical constraints of the form $X_i = f(X_{i_1}, \ldots, X_{i_r})$. In particular, linear constraints are functional

in all variables. A constraint network with sufficient functional constraints has the property for which we are looking.

In the second case we use a structure from combinatorial mathematics called a Turan set. By making sure that constraints with sufficiently tight relations have scopes which form a Turan set we can find a polynomial bound on the number of partial solutions to any subset of the variables.

The paper is structured as follows. In Sect. 2 we introduce the necessary basic definitions and give a motivating example. Section 3 is a study of different forms of functional CSP instances. Section 4 presents a novel tractable class which guarantees a polynomial bound on the number of solutions, but which is not based on any form of functionality.

2 Background

We first define the problem we are trying to solve.

Definition 1. *A **constraint network** is a triple $\langle V, D, C\rangle$ where:*

- *V is a finite set of variables;*
- *D is a finite domain;*
- *C is a set of constraints. Each **constraint** $c \in C$ consists of a pair $c = \langle \sigma(c), \rho(c)\rangle$ where $\sigma(c) \in V^*$, the constraint **scope**, is a list of variables and $\rho(c) \subseteq D^{|\sigma(c)|}$, is the constraint **relation**.*

*A **solution** to $P = \langle V, D, C\rangle$ is a mapping $s : V \to D$ which satisfies each constraint. That is, for each $\langle \sigma, \rho\rangle \in C$, $s(\sigma) \in \rho$.*

*For any set of variables $X \subseteq V$ we have the standard notion of the **induced network** $P[X] = \langle X, D, C'\rangle$ on X, where, for every $c \in C$ whose scope includes at least one variable of X there is a corresponding induced constraint $c[X] \in C'$. The scope of $c[X]$ is the sublist of variables of σ that occur in X and the relation of $c[X]$ consists of those tuples of values that extend to tuples in ρ.*

2.1 Ordered Polynomial Tractability

We are interested in constraint networks for which we only ever generate a polynomial number of partial solutions during complete

backtrack search. As such we are interested in networks which have particularly well behaved variable orderings.

Definition 2. *A class of constraint networks is* ***ordered polynomial*** *if there is some polynomial p such that, for any such instance P, there is some ordering $x_1 < x_2 < \ldots < x_n$ of the variables of P where, for each $i = 1, \ldots, n$ the induced network $P[\{x_1, \ldots, x_i\}]$ has at most $p(|P[\{x_1, \ldots, x_i\}]|)$ solutions.*

Example 1. This example describes the tractable class of constraint networks with fractional edge cover number at most k discovered by Grohe and Marx [9].

For any constraint network $P = \langle V, D, C\rangle$ we define the structure of P to be the hypergraph $H(P)$ with vertex set V and a hyperedge for each constraint scope (abstracted as a set of variables).

A fractional edge cover of the hypergraph $\langle V, E\rangle$ is a mapping $\psi : E \to [0, \infty)$ such that, for every $v \in V$, $\sum_{e \in E, v \in e} \psi(e) \geq 1$. The weight of ψ is $\sum_{e \in E} \psi(E)$ and the fractional weight, $\rho^* H$, of H is the minimum weight of all fractional edge covers of H.

Grohe and Marx [9] proved that the number of solutions to any constraint network P is at most $|P|^{\rho^*(H(P))}$.

Since the fractional edge cover number of any induced network is at most that of the original network it follows that the class of constraint networks with fractional edge cover number at most k is ordered polynomial.

Grohe and Marx proved that enumerating all solutions can be done in time $|I|^{\rho^*(H(I))+O(1)}$. We generalise this result to arbitrary ordered polynomial classes.

Proposition 1. *Let $P = \langle V, D, C\rangle$ be any constraint network with variable ordering $x_1 < x_2 < \ldots < x_n$ such that the induced network $P[\{x_1, \ldots, x_i\}]$ can be solved in time $p(|P[\{x_1, \ldots, x_i\}]|)$. All solutions to P can be enumerated in time $p(|P|).|P|^2$.*

Proof. For $i = 1, \ldots, |V|$ the algorithm generates a list L_i of solutions to $P[\{x_1, \ldots, x_i\}]$. The list L_1 contains at most $|D|$ solutions. Since every solution in L_{i+1} induces a solution in L_i by projection, to find L_{i+1} we have only to find those solutions in L_i which extend to a solution of $P[\{x_1, \ldots, x_{i+1}\}]$. Clearly this extension can be done in time $|L_i|.|D|.|P[\{x_1, \ldots, x_{i+1}\}]|$.

Now since $|P[\{x_1, \ldots, x_{i+1}\}]| \geq |P[\{x_1, \ldots, x_i\}]|$ we can bound the total running time by $|V|.|D|.p(|P|).|P| \leq p(|P|).|P|^2$.

However, a network with fractional edge cover number at most k must include a constraint whose scope includes at least $1/k$ of all variables. We would like to find tractable ordered polynomial classes which do not require such unreasonably large scopes.

3 Functional Constraint Networks

In this section, we begin by studying the class of constraint networks which have sufficient functional constraints to guarantee backtrack free search.

Functional constraints have been extensively studied in the case of *binary* networks [6, 4, 5]. We extend previous studies to the case of functional constraints of arbitrary arity and generalise to the case of incrementally-functional networks.

Definition 3. *A constraint $\langle \sigma, \rho \rangle$ is **functional** on variable $i \in \sigma$ if ρ contains no two assignments y, z differing only at variable i.*

As an example of a functional constraint, let ρ be any relation such that every pair of tuples in ρ differ on at least two variables. This is true of all linear relations, such as arbitrary Boolean relations to which a parity bit has been added. Then the constraint $\langle \sigma, \rho \rangle$ is functional on every variable $i \in \sigma$.

Definition 4. *A constraint network P is **functional** if there exists a variable ordering $x_1 < x_2 < \ldots < x_n$ such that, for all $i \in \{1, \ldots, n\}$, there is some constraint of $P[\{x_1, \ldots, x_i\}]$ that is functional on x_i.*

It is easy to see that a functional network has at most one solution. Furthermore, it is not difficult to show that functional networks form a tractable class: they are both identifiable and solvable in polynomial time. We will, in fact, prove this for the much larger class defined below.

Definition 5. *A constraint network P is **incrementally functional** if there is an ordering $x_1 < x_2 < \ldots < x_n$ of its variables such that for all $i \in \{1, \ldots, n-1\}$, each solution to $P[\{x_1, \ldots, x_i\}]$ extends to at most one solution to $P[\{x_1, \ldots, x_{i+1}\}]$.*

An incrementally functional constraint network has at most one solution.

Proposition 2. *The number of nodes in the backtracking search tree of an incrementally functional constraint network is $O(n)$ when the variables are instantiated according to the correct ordering.*

It is possible to determine in polynomial time whether a network P is incrementally functional. This is a corollary of the following result which says that we can determine in polynomial time the maximum-cardinality subset $M \subseteq \{1, \ldots, n\}$ such that $P[M]$ is incrementally functional. This provides us with a simple variable-ordering heuristic: if the variables in M are instantiated in the order that makes $P[M]$ incrementally functional, then, in the backtracking search tree, there is no branching at any of the variables of M.

Proposition 3. *Given a constraint network $P = \langle V, D, C \rangle$, it is possible to find in polynomial time the maximum-cardinality set $M \subseteq V$ such that $P[M]$ is incrementally functional.*

Proof. We initialize M to the empty set and a_M to the empty tuple. We then use a greedy algorithm to repeatedly add variables i to M if there is *at most* one value a_i for i which is consistent with the partial assignment a_M. If such an a_i exists then a_M is extended to include the assignment of a_i to i. If a_M has no consistent extension to variables $M \cup \{i\}$, then we halt, returning V (since, in this case, the conditions of Definition 5 are satisfied on i and so trivially on all variables not in $M \cup \{i\}$).

Consider the set M returned by this greedy algorithm. Suppose that $P[M']$ is incrementally functional where $|M'| > |M|$. Without loss of generality, we can assume that $M' = \{1, \ldots, t\}$ and that the variable ordering which makes P incrementally functional on M' is the usual ordering of the integers. Let $i = \min(M' \setminus M)$ and let $X = \{j \in M' : j < i\}$. Since $i \in M'$, all consistent assignments to the variables P can be extended to at most one consistent assignment to the variables $X \cup \{i\}$. But, by choice of i, $X \subseteq M$, and hence all consistent assignments to the variables M can be extended to at most one consistent assignment to the variables $M \cup \{i\}$. Thus, i would have been added to M by our greedy algorithm. This contradiction completes the proof.

Corollary 1. *It is possible to determine in polynomial time whether a constraint network is incrementally functional.*

For the constraint network $P = \langle V, D, C \rangle$, if the induced network $P[M]$ is incrementally functional, then after instantiating the variables in $V - M$, P becomes incrementally functional. The converse is not necessarily true, in the sense that P may be incrementally functional on all remaining variables after instantiation of the variables in a proper subset of $V - M$. This leads us to notions which are related to strong backdoor sets.

In a SAT instance, given a polynomial-time algorithm A (such as unit-propagation), a set X of variables is a **strong backdoor set** with respect to A if, for any assignment to the variables in X, the algorithm A determines whether or not this assignment can be extended to a complete solution. The set of problem instances having a strong backdoor set of size $O(\log n)$ is a hybrid tractable class [17], but does not provide a polynomial bound on the number of solutions. Szeider [15] showed that when the SAT algorithm A is unit propagation, pure literal elimination or a combination of both of these, the detection of a strong backdoor set (with respect to A) of size bounded by a fixed integer k is W[P]-complete. This means that it is highly unlikely that smallest backdoor sets can be found more efficiently than by exhaustive search.

The remainder of this section is devoted to types of backdoor sets, called *root sets*, defined in terms of simple forms of functionality.

Definition 6. *In a constraint network $P = \langle V, D, C \rangle$, a **root set** is a subset Q of the variables for which there exists a variable ordering $x_1 < x_2 < \ldots < x_n$ such that, for all $i \in V - Q$, there is some constraint of $P[\{x_1, \ldots, x_i\}]$ that is functional on x_i.*

The existence of a root set Q means that the CSP instance I will become functional after instantiation of all variables in Q. It is therefore of interest to find a minimum-cardinality root set. David [5] showed that this can be achieved in polynomial time in the case of binary CSPs. Unfortunately, if I contains ternary functional constraints, then finding the minimum-cardinality root set is NP-hard, as we now show.

Theorem 1. *The problem of finding a minimum-cardinality root set in a ternary constraint network is NP-hard, for all $d \geq 2$ (where d is the maximum domain size).*

Proof. We demonstrate a polynomial reduction from MAX CLIQUE (which is known to be NP-complete [12]). Let G be a graph with n vertices $1, \ldots, n$ and m edges. For simplicity of presentation, we identify a clique C in G with its vertex set. We will construct a ternary constraint network P_G such that $\{X_i : i \notin C\} \cup \{X_0\}$ is a minimum-cardinality root set of P_G if and only if C is a maximum clique in G.

We first give an example of a ternary functional constraint which is functional on only one of its variables. Let R_3 denote the relation $\{\langle 0,0,0\rangle, \langle 0,1,0\rangle, \langle 1,0,0\rangle, \langle 1,1,1\rangle\}$. It is easy to verify that R_3 is functional only on its third variable.

The constraint network P_G has a "dummy" variable X_0 (which, by construction, must appear in every root set) as well as a variable X_i corresponding to each of the n vertices of G. Apart from these variables X_i $(i = 0, \ldots, n)$, P_G has two other types of variables: non-edge variables denoted by Y_i $(1 \leq i \leq M$ where $M = \frac{n}{2}(n-1) - m)$ and cascade variables Z_j^i $(i = 1, \ldots, n;\ 1 \leq j \leq N)$. For each pair of distinct vertices $j, k \in \{1, \ldots, n\}$ which are not connected by an edge in G, there is a corresponding non-edge variable Y_i in P_G together with two ternary functional constraints: $\langle\langle X_0, X_j, Y_i\rangle, R_3\rangle$ and $\langle\langle X_0, X_k, Y_i\rangle, R_3\rangle$. These constraints are both functional on Y_i and are the only constraints of P_G which are functional on Y_i.

For each of the X_i variables, there is a cascade of ternary functional constraints from the set of *all* the non-edge variables Y_j $(1 \leq j \leq M)$ to X_i. This is illustrated in Fig. 1 which shows a cascade from the variables $Y_1, \ldots, Y_8$ to the variable X_i via the cascade variables $Z_1^i, \ldots, Z_6^i$. Each two-tailed arrow represents a ternary functional constraint from the two tail variables to the head variable: if U, V are the tail variables and W the head variable, then there is the constraint $\langle\langle U, V, W\rangle, R_3\rangle$ in P_G. We require a total of $N = n(2^{\lceil \log_2 M \rceil} - 2) = O(n^3)$ cascade variables.

For any subset $C \subseteq \{1, \ldots, n\}$ of the vertices of G, denote by R_C the set $\{X_i : i \notin C\} \cup \{X_0\}$. C is a clique if and only if R_C contains one of X_j, X_k for each non-edge $\{j, k\}$ in G. We claim that,

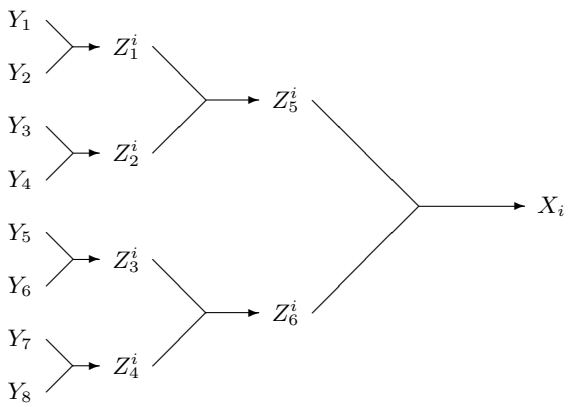

Fig. 1. Example of a cascade of variables from the variables $Y_1, \ldots, Y_8$ to variable X_i.

by construction of P_G, this is a necessary and sufficient condition for R_C to be a root set of P_G. For each non-edge $\{j, k\}$ with corresponding variable Y_i in P_G, the only functional constraints on Y_i are $\langle\langle X_0, X_j, Y_i\rangle, R_3\rangle$ and $\langle\langle X_0, X_k, Y_i\rangle, R_3\rangle$; hence R_C is a root set only if it contains one of X_j, X_k. If R_C contains one of X_j, X_k for each non-edge $\{j, k\}$ in G, then any ordering which places the variables in R_C first, then the variables $\{Y_i : 1 \leq i \leq M\}$, then the variables $\{Z_j^i : 1 \leq i \leq n,\ \ 1 \leq j \leq N\}$ and finally the variables $\{X_i : i \leq i \leq n\} - R_C$ satisfies the conditions of Definition 6.

It is possible to replace some number of the X_i variables ($1 \leq i \leq n$) in the root set R_C by some number of the Y_j and Z_k^i variables (for example, X_i in Fig. 1 could be replaced by Z_5^i, Z_6^i), but, by our construction, this never reduces the cardinality of the root set. We can conclude that R_C is a minimum-cardinality root set of P_G if and only if C is a maximum clique in G. This reduction from MAX CLIQUE is clearly polynomial.

4 Turan Sets

In this section, we study conditions which guarantee the existence of only a polynomial number of solutions, but which are not so restrictive as to guarantee the existence of at most one solution. Very little work appears to have done in this area, although David [4]

showed that constraint networks which have a functional constraints for every possible arity-k scope are solvable in polynomial time. We will generalise this result here, but first we show that many weak constraints are sufficient to control the number of solutions.

Definition 7. *For any domain D we define a **domain pair arity** function to be a symmetric mapping $\alpha : D^2 \to \mathbb{N}^+$.*

The size of α is then defined as $|\alpha| = \sum_{\{a,b\} \subseteq D} \alpha(a,b)$.

*For any $\{a,b\} \subseteq D$, we define the **reduction** of α at $\{a,b\}$ to be*

$$\alpha^{\{a,b\}-}(x,y) = \begin{cases} \alpha(x,y) - 1 & \text{if } \{x,y\} = \{a,b\}, \\ \alpha(x,y) & \text{otherwise.} \end{cases}$$

*For any domain pair arity function α we say that constraint network $P = \langle V, D, C \rangle$ is α**-restrictive** if*

$$\forall \{a,b\} \subseteq D, U \subseteq V, |U| = \alpha(a,b), \exists \langle \sigma, \rho \rangle \in C, \sigma \in U^* \text{ and } [a,b]^{|\sigma|} \not\subseteq \rho \ .$$

Let $F(n, D, \alpha)$ be the maximum number of solutions to any α-restrictive constraint network with domain D and n variables.

Theorem 2. $F(n, D, \alpha) \leq n^{|\alpha|}$.

Proof. Let D be any domain and α be any domain pair arity function. Choose an n variable α-restrictive constraint network $P = \langle V, D, C \rangle$ with $F(n, D, \alpha)$ solutions.

Choose any variable $x \in V$. We say that a pair of solutions to P are an $\{a,b\}$-pair at x if they differ only in their value for variable x: one having value a, the other value b.

Let $S_{\{a,b\}}$ be the solutions to P which are in $\{a,b\}$-pairs. Let S_1 be the solutions to P that are in no $\{a,b\}$-pair for any $\{a,b\} \subseteq D$.

We get an upper bound on the number of solutions to P, $F(n, D, \alpha)$, by counting the size of these sets.

First, for any $\{a,b\}$ where $\alpha(a,b) > 1$, consider the set $S_{\{a,b\}}$.

Choose any $U \subseteq V$ with $x \in U$ and $|U| = \alpha(a,b)$.

Since P is α-restrictive there is a constraint $\langle \sigma, \rho \rangle \in C$ for which $\sigma \subseteq U^*$ and $[a,b]^{|\sigma|} \not\subseteq \rho$. Hence the restriction of $S_{\{a,b\}}$ to $U - \{x\}$ cannot contain $[a,b]^{|U|-1}$.

So, for each such U, we can build a constraint with scope $U - \{x\}$ that disallows no solution of $S_{\{a,b\}}$, but whose relation does not

contain $[a,b]^{|U|-1}$. Add all such constraints to the induced network $P[V-\{x\}]$ to obtain the $\alpha^{\{a,b\}-}$-restrictive network $P(a,b)$. By construction, every solution in $S_{\{a,b\}}$ restricted to $V-\{x\}$ is a solution to $P(a,b)$. Since every element of $S_{\{a,b\}}$ has value either a or b at x we know that the number of solutions to $P(a,b)$ is at least half of the size of $S_{\{a,b\}}$.

Now consider the set S_1. Each solution to $P[V-\{x\}]$ extends to at most one element of S_1, so the α-restrictive network $P[V-\{x\}]$ has precisely $|S_1|$ solutions that extend to an element of S_1.

Finally, observe that if $\alpha(a,b)=1$ then $S_{\{a,b\}}$ is empty since there is a constraint with scope $\langle x\rangle$ whose relation does not contain $[a,b]$.

We have shown that:

$$F(n,D,\alpha) \leq \left(\sum_{\{a,b\}\subseteq D, \alpha(a,b)>1} 2F(n-1,D,\alpha^{\{a,b\}-}) \right) + F(n-1,D,\alpha) \ . \tag{1}$$

We can now prove the theorem by induction on $|\alpha|$.

The base case is when α is identically 1. Here $F(n,D,\alpha)=1$ since in the solution set to any α-restrictive constraint network we can have, for each variable, at most one of each pair of domain values.

For induction, and using inequality (1), we can assume that

$$F(n,D,\alpha) \leq \binom{D}{2}(n-1)^{|\alpha|-1} + (n-1)^{|\alpha|} \ .$$

It remains to show that

$$n^{|\alpha|} \geq \binom{D}{2}(n-1)^{|\alpha|-1} + (n-1)^{|\alpha|}.$$

We can rewrite this inequality as:

$$\left(1+\frac{1}{n-1}\right)^{|\alpha|} \geq \frac{\binom{D}{2}}{(n-1)} + 1 \ .$$

This equality must hold since $|\alpha| \geq \binom{D}{2}$.

Given an arbitrary domain pair arity function α it is clear that if P is α-restrictive then so is every network induced by P on a subset of its variables. It follows immediately from Theorem 2 and Proposition 1 that the class of α-restrictive constraint networks is polynomially solvable for a fixed domain size.

Corollary 2. *For any fixed domain D and domain pair arity function α the class of α-good constraint networks over domain D is polynomial time solvable.*

Now we will need the notion of a Turan set in order to define classes of α-restrictive networks for a constant function α. This will allow us not only to give some concrete examples of tractable classes but also to estimate the minimum number of constraints in an α-restrictive constraint network.

Definition 8. *We say that a subset of variables σ **represents** another set τ if σ is contained in τ.*

*An (n,k)-**Turan system** is a pair $\langle \chi, \mathcal{B} \rangle$ where $\mathcal{B}$ is a collection of subsets of the n-element set χ such that every k-element subset of χ is represented by some set in $\mathcal{B}$. The size of the system $\langle \chi, \mathcal{B} \rangle$ is the number of subsets in $\mathcal{B}$.*

The restricted notion of a Turan system where every member is required to have precisely $r < k$ elements has been well-studied in the mathematics community and, for many set of parameters, minimal size examples are known [14, 16].

Definition 9. *An n-variable constraint network over domain D is said to be k-**Turan** if the scopes of the constraints $\langle \sigma, \rho \rangle$ for which:*

$$\forall a, b \in D, [a,b]^{|\sigma|} \not\subseteq \rho$$

are an (n,k)-Turan system.

We immediately obtain the following result from Theorem 2.

Theorem 3. *For any domain D and fixed k, the class of k-Turan constraint networks is tractable.*

Proof. Let P be any k-Turan constraint network over domain D.

Since every k-set of variables contains the scope $\langle \sigma, \rho \rangle$ of a constraint for which:

$$\forall a, b \in D, [a,b]^{|\sigma|} \not\subseteq \rho$$

we immediately get that P is α-restrictive for the constant domain pair arity function $\alpha(a,b) = k$, and so the class is polynomially solvable.

To see that such a class is polynomially recognisable observe that there are polynomially many subsets of variables of size k and, for each of these sets we have only to check the relations of constraints whose scope they contain. That is, one check is required for each pair of domain elements, for each constraint and for each subset of size k giving time complexity $O(|D|^2|P|^{k+1})$.

To give a concrete example, the minimum size of a $(n, s+1)$-Turan set of two element subsets is known to be $\frac{n}{2}(\frac{n}{s}-1)$ when n is a multiple of s. Although this compares well with the number $\frac{n}{2}(\frac{n}{2}-1)$ of size-2 subsets of $\{1,\ldots,n\}$, it indicates that we may still require $\Theta(n^2)$ binary constraints for Theorem 3 to imply a polynomial number of solutions.

It is worth observing that the restriction:

$$\forall a, b \in D, [a,b]^{|\sigma|} \not\subseteq \rho$$

is not very strong. For instance every clause (seen as a constraint over a Boolean domain) satisfies the restriction. There is just one domain pair $0, 1$ and so we only require there to be at least one disallowed tuple.

Hence, a direct consequence of Theorem 3 is for the case of k-SAT (boolean domains, where each constraint is a clause on k-variables).

Corollary 3. *The class of k-SAT instances where every k-tuple is restricted by a k-clause is tractable.*

What is more, every functional constraint satisfies this property.

Proposition 4. *Let $\langle \sigma, \tau \rangle$ be any functional constraint. We have that:*

$$\forall a, b \in D, [a,b]^{|\sigma|} \not\subseteq \rho \ .$$

Proof. Suppose that $\langle \sigma, \tau \rangle$ is functional at x. Choose arbitrary domain elements a and b. We know that any assignment of values from $\{a, b\}$ to the variables of σ other than x extends to at most one value at x. So we are done.

The result of David's [4] cited in the introduction to this section is therefore another corollary of Theorem 3.

Corollary 4. *If a constraint network P has all arity-q constraints and each of these constraints is functional then we can solve I in polynomial time*

The class of k-Turan constraint networks is quite large. Many of the well-known global constraints [1] also satisfy the rather weak restriction, but we have only space to give the (simple) proofs for functional constraints and Boolean clauses. The k-Turan constraint networks can be seen as requiring sufficiently many small scopes with reasonable constraint relations - in contrast to networks of bounded fractional edge cover number [9] which require sufficient coverage by quite large scopes.

5 Conclusion

We have defined different classes of constraint networks whose tractability stems from the fact that each network has only a polynomial number of solutions. This means that we can list or count all solutions or find all optimal solutions (according to any polytime objective function) in polynomial time.

Incrementally functional constraint networks have a single solution and a forward-checking search tree is linear when a dynamic smallest-domain first variable-ordering is used. Finding a maximum-cardinality subset of the variables on which a network is incrementally functional is polynomial-time. Finding a maximum-cardinality root set (a set of variables on which all others are functionally dependent) would appear to be NP-hard except in the case that each variable is functionally dependent on a single previous variable.

We have also presented novel tractable classes of constraint networks with *no* functional constraints which have a polynomial bound on the number of solutions. These k-Turan networks require many constraints with small scopes but put only a very weak restriction on the relations of these constraints. As such this is an interesting contrast with the (structural) class of networks with small fractional edge cover number.

An interesting open question is whether there exist other conditions guaranteeing a polynomial number of solutions.

References

1. Nicolas Beldiceanu, Mats Carlsson, and Jean-Xavier Rampon. Global constraint catalog. 2nd edition. Technical Report T2010:07, Swedish Institute of Computer Science, SICS, Isafjordsgatan 22, Box 1263, SE-164 29 Kista, Sweden, November 2010.
2. A.A. Bulatov. A dichotomy theorem for constraints on a three-element set. In *Proceedings 43rd IEEE Symposium on Foundations of Computer Science, FOCS'02*, pages 649–658. IEEE Computer Society, 2002.
3. Martin C. Cooper, Peter G. Jeavons, and András Z. Salamon. Generalizing constraint satisfaction on trees: Hybrid tractability and variable elimination. *Artif. Intell.*, 174(9-10):570–584, 2010.
4. Philippe David. *Prise en compte de la sémantique dans les problèmes de satisfaction de contraintes : étude des contraintes fonctionnelles.* PhD thesis, LIRMM, Université Montpellier II, 1994.
5. Philippe David. Using pivot consistency to decompose and solve functional csps. *J. Artif. Intell. Res. (JAIR)*, 2:447–474, 1995.
6. Yves Deville and Pascal Van Hentenryck. An efficient arc consistency algorithm for a class of csp problems. In *Proceedings of the 12th international joint conference on Artificial intelligence - Volume 1*, pages 325–330, San Francisco, CA, USA, 1991. Morgan Kaufmann Publishers Inc.
7. 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.
8. M. Grohe. The structure of tractable constraint satisfaction problems. In *MFCS*, pages 58–72, 2006.
9. Martin Grohe and Dániel Marx. Constraint solving via fractional edge covers. In *SODA*, pages 289–298. ACM Press, 2006.
10. Peter G. Jeavons, David A. Cohen, and Marc Gyssens. A test for tractability. In Eugene C. Freuder, editor, *CP*, volume 1118 of *Lecture Notes in Computer Science*, pages 267–281. Springer, 1996.
11. P. Jegou. Decomposition of domains based on the micro-structure of finite constraint-satisfaction problems. In *Proceedings of the 11th National Conference on Artificial Intelligence*, pages 731–736, Menlo Park, CA, USA, jul 1993. AAAI Press.
12. R. M. Karp. Reducibility Among Combinatorial Problems. In R. E. Miller and J. W. Thatcher, editors, *Complexity of Computer Computations*, pages 85–103. Plenum Press, 1972.
13. András Z. Salamon and Peter G. Jeavons. Perfect constraints are tractable. In *Proceedings of the 14th International Conference on Principles and Practice of Constraint Programming, CP 2008, Sydney, Australia, 14–18 September*, volume 5202 of *Lecture Notes in Computer Science*, pages 524–528. Springer, 2008.
14. A. F. Sidorenko. Precise values of turan numbers. *Mathematical Notes*, 42:913–918, 1987. 10.1007/BF01137440.
15. Stefan Szeider. Backdoor sets for dll subsolvers. *J. Autom. Reasoning*, 35(1-3):73–88, 2005.
16. P. Turan. Research problems. *Publ. Hung. Acad. Sci.*, 6:417–423, 1961.
17. Ryan Williams, Carla P. Gomes, and Bart Selman. Backdoors to typical case complexity. In Georg Gottlob and Toby Walsh, editors, *IJCAI*, pages 1173–1178. Morgan Kaufmann, 2003.