A Microstructure-based Family of Tractable Classes for CSPs*

Martin C. Cooper[1], Philippe Jégou[2], and Cyril Terrioux[2]

[1] IRIT
University of Toulouse III
31062 Toulouse, France
cooper@irit.fr

[2] Aix-Marseille Université, CNRS, ENSAM, Université de Toulon, LSIS UMR 7296
Avenue Escadrille Normandie-Niemen
13397 Marseille Cedex 20, France
{philippe.jegou, cyril.terrioux}@lsis.org

Abstract. The study of tractable classes is an important issue in Artificial Intelligence, especially in Constraint Satisfaction Problems. In this context, the Broken Triangle Property (BTP) is a state-of-the-art microstructure-based tractable class which generalizes well-known and previously-defined tractable classes, notably the set of instances whose constraint graph is a tree. In this paper, we propose to extend and to generalize this class using a more general approach based on a parameter k which is a given constant. To this end, we introduce the k-BTP property (and the class of instances satisfying this property) such that we have 2-BTP = BTP, and for $k > 2$, k-BTP is a relaxation of BTP in the sense that k-BTP $\subsetneq$ $(k+1)$-BTP. Moreover, we show that if k-TW is the class of instances having tree-width bounded by a constant k, then k-TW $\subsetneq$ $(k+1)$-BTP. Concerning tractability, we show that instances satisfying k-BTP and which are strong k-consistent are tractable, that is, can be recognized and solved in polynomial time. We also study the relationship between k-BTP and the approach of Naanaa who proposed a set-theoretical tool, known as the directional rank, to extend tractable classes in a parameterized way. Finally we propose an experimental study of 3-BTP which shows the practical interest of this class, particularly w.r.t. the practical solving of instances satisfying 3-BTP and for other instances, w.r.t. to backdoors based on this tractable class.

1 Introduction

Finding islands of tractability, generally called *tractable classes* is an important issue in Artificial Intelligence, especially in Constraint Satisfaction Problems (CSPs [1]). Many studies have addressed this issue, from the very beginnings of Artificial Intelligence. These results are often theoretical in nature with, in certain cases, tractable classes which can be considered as somewhat artificial. But some tractable classes have actually been used in practice, such as the classes defined by constraint networks with bounded tree-width [2, 3]. More recently, the concept of hybrid class has been defined,

* supported by ANR Project ANR-10-BLAN-0210 and EPSRC grant EP/L021226/1.

for example with the class BTP [4]. This class strictly contains both structural tractable classes (such as tree-structured CSPs) and tractable classes defined by language restrictions. One major advantage of this class, in addition to its generalization of already-known tractable classes, is related to its practical interest. Indeed, instances of this class can be solved in polynomial time using algorithms, such as MAC (Maintaining Arc-Consistency [5]) and RFL (Real Full Look-ahead [6]), implemented in efficient solvers which allows it to be used directly in practice. In addition, it may also help to explain theoretically the practical efficiency of solvers, even though the theoretical complexity of the algorithms employed by the solvers is exponential in the worst case.

In this paper, we return to this type of approach by generalizing the tractable class BTP which is defined by a property excluding certain patterns (called *Broken Triangles*) in the microstructure graph associated with a binary CSP instance. Very recent work in this same direction introduced the class ETP [7] which generalizes BTP by relaxing some of its conditions, since it tolerates some broken triangles which are forbidden by BTP. Here we propose a broader generalization called k-BTP which extends this previous work along two axes. First, in the spirit of ETP, the new class allows the presence of a larger number of broken triangles, generalizing strictly ETP (and thus BTP). Secondly, the class k-BTP is parameterized by a constant k, thus providing a generic version, which may prove of theoretical interest for general values of k, although in practice we consider the case $k = 3$ to be of most interest. Thus, while BTP is defined for sets of three variables and ETP for sets of four variables, k-BTP is defined on the basis of sets of $k + 1$ variables where k is a fixed constant. According to this approach, $BTP = 2$-BTP while $ETP \subsetneq 3$-BTP. Thus, this approach makes it possible to strictly generalize these two classes. Furthermore, k-BTP retains some of their interesting properties and practical advantages mentioned above. Notably, we show that classical algorithms such as MAC or RFL can solve instances belonging to k-BTP in polynomial time, assuming that these instances verify *Strong k-Consistency* [8]. Moreover, we highlight the relationships of this class with known structural and hybrid classes. We show in particular that the class of constraint networks whose tree-width is bounded by k is strictly included in the class k-BTP. This result gives a first answer to a question recently asked by M. Vardi about the relationships between ETP and the tractable class induced by instances of bounded tree-width. We also highlight a recent but relatively unknown result that was proposed by Naanaa [9] whose relationships with k-BTP we investigate.

In Section 2 we recall the definitions of the tractable classes BTP and ETP. In Section 3 we define the new class k-BTP and show that instances from this class can be detected in polynomial time even when the variable order is not known in advance. Furthermore, we show that, under the extra hypothesis of strong k-consistency, such instances can be solved in polynomial time, and, in fact, standard algorithms will solve them. In Section 4 we investigate relationships between k-BTP and several known tractable classes and in Section 5 we report results of experimental trials on benchmark problems.

2 Background

Formally, a *constraint satisfaction problem* also called *constraint network* is a triple (X, D, C), where $X = \{x_1, \ldots, x_n\}$ is a set of n variables, $D = (D_{x_1}, \ldots, D_{x_n})$ is a list of finite domains of values, one per variable, and $C = \{c_1, \ldots, c_e\}$ is a finite set of e constraints. Each constraint c_i is a pair $(S(c_i), R(c_i))$, where $S(c_i) = \{x_{i_1}, \ldots, x_{i_k}\} \subseteq X$ is the *scope* of c_i, and $R(c_i) \subseteq D_{x_{i_1}} \times \cdots \times D_{x_{i_k}}$ is its *compatibility relation*. The *arity* of c_i is $|S(c_i)|$. In this paper, we only deal with the case of binary CSPs, that is CSPs for which all the constraints are of arity 2. Hence, we will denote by c_{ij} the constraints involving x_i and x_j. The structure of a constraint network is represented by a graph, called the *constraint graph*, whose vertices correspond to variables and edges to the constraint scopes. An assignment to a subset Y of X is said to be *consistent* if it does not violate any constraint whose scope is included in Y. We use the notation $R(c_{ij})[a]$ to represent the set of values in D_{x_j} compatible with $a \in D_{x_i}$. Thus, if there is a constraint with scope $\{i, j\}$, then $R(c_{ij})[a] = \{b \in D_{x_j} | (a, b) \in R(c_{ij})\}$; if there is no constraint with scope $\{i, j\}$, then, by default, $R(c_{ij})[a] = D_{x_j}$. We recall the BTP property presented in [4].

Definition (BTP) A binary CSP instance (X, D, C) satisfies the *Broken Triangle Property* (BTP) w.r.t. the variable ordering $<$ if, for all triples of variables (x_i, x_j, x_k) s.t. $i < j < k$, if $(v_i, v_j) \in R(c_{ij})$, $(v_i, v_k) \in R(c_{ik})$ and $(v_j, v'_k) \in R(c_{jk})$, then either $(v_i, v'_k) \in R(c_{ik})$ or $(v_j, v_k) \in R(c_{jk})$. If neither of these two tuples exist, (v_i, v_j, v_k, v'_k) is called a *broken triangle on x_k w.r.t. x_i and x_j*.

If there exists at least one broken triangle on x_k w.r.t. x_i and x_j, (x_i, x_j, x_k) is called a *broken triple on x_k w.r.t. x_i and x_j*. Let BTP be the set of the instances for which BTP holds w.r.t. some variable ordering. The BTP property is related to the compatibility between domain values, which can be graphically visualized (Figure 1) on the microstructure graph. For example, in Figure 1 (a), there is a broken triangle on x_3 with respect to the variables x_1 and x_2 since we have $(v_1, v'_3) \notin R(c_{13})$ and $(v_2, v_3) \notin R(c_{23})$ while $(v_1, v_2) \in R(c_{12})$, $(v_1, v_3) \in R(c_{13})$ and $(v_2, v'_3) \in R(c_{23})$ hold. So (x_1, x_2, x_3) is a broken triple on x_3 w.r.t. x_1 and x_2. In contrast, in Figure 1 (b), if one of the two dashed edges (that is binary tuples) appears in the microstructure, the BTP property holds for all variable orderings.

Very recently, the property BTP has been relaxed to the Extendable-Triple Property [7] by considering four variables rather than three, and allowing some broken triangles.

Definition (ETP) A binary CSP instance P satisfies the *Extendable-Triple Property* (ETP) with respect to the variable ordering $<$ if, and only if, for all subsets of four variables (x_i, x_j, x_k, x_l) such that $i < j < k < l$, there is at most one broken triple on x_l among (x_i, x_j, x_l), (x_i, x_k, x_l) and (x_j, x_k, x_l).

In this way, a binary CSP can satisfy the ETP property while it contains two broken triples among (x_i, x_j, x_k, x_l), one on x_k, and another one on x_l, while none is possible with BTP. So, ETP strictly generalizes BTP since each instance satisfying BTP satisfies ETP while the reverse is false. So the class of instances satisfying BTP (denoted

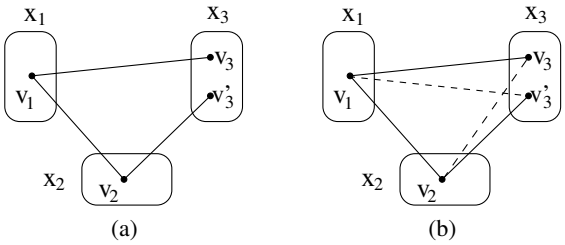

Fig. 1. A non-BTP instance (a) and a BTP one (b) w.r.t. the order $x_1 < x_2 < x_3$ if one of the dashed lines occurs.

BTP) is strictly included in the class of instances satisfying ETP (denoted ETP) as indicated in Theorem 1 of [7] ($BTP \subsetneq ETP$). As in the case of BTP, ETP allows us to define a tractable class but we need to impose an additional property related to the level of local consistency which must be verified. While the set of instances satisfying BTP define a tractable class, the set of instances satisfying ETP must also satisfy *Strong-Path-Consistency* [8], that is arc and path-consistency. Nevertheless, such instances have some of the desirable properties of instances satisfying BTP, e.g. they can be solved in polynomial time by usual algorithms such as MAC or RFL. In the next section, we introduce a new property which generalizes BTP and ETP.

3 k-BTP: Definition and Properties

In this section, we introduce a new property k-BTP which generalizes previous work along two axes. First, the property ETP is relaxed in the sense that we allow more broken triangles than ETP when considering subsets of four variables. But we also introduce a parameter $k \geq 2$ allowing us to consider subsets of $k+1$ variables, with $k = 2$ corresponding to BTP and $k = 3$ corresponding to a strict generalization of ETP.

Definition (k-BTP) *A binary CSP instance P satisfies the property k-BTP for a given k ($2 \leq k < n$) and with respect to the variable ordering $<$ if, and only if, for all subsets of $k+1$ variables $x_{i_1}, x_{i_2}, \ldots x_{i_{k+1}}$ such that $i_1 < i_2 < \ldots < i_{k-1} < i_k < i_{k+1}$, there is at least one triple of variables $(x_{i_j}, x_{i_{j'}}, x_{i_{k+1}})$ with $1 \leq j \neq j' \leq k$ such that there is no broken triangle on $x_{i_{k+1}}$ w.r.t. x_{i_j} and $x_{i_{j'}}$. Let k-BTP be the set of the instances for which k-BTP holds w.r.t. some variable ordering.*

One can observe that 2-BTP is exactly BTP while 3-BTP includes ETP. So, we can immediately extend Theorem 1 of [7] since $BTP \subsetneq ETP \subsetneq 3\text{-}BTP$. But above all, a more general result holds, which is an immediate consequence of the definition of k-BTP:

Theorem 1 *For all $k \geq 2$, k-$BTP \subsetneq (k{+}1)$-BTP*

To analyze the tractability of k-BTP, we now show that the instances of this class can be recognized in polynomial time:

Theorem 2 *Given a binary CSP instance $P = (X, D, C)$ and a constant k with $2 \leq k < n$, there is a polynomial time algorithm to find a variable ordering $<$ such that P satisfies k-BTP w.r.t. $<$, or to determine that no such ordering exists.*

Proof: As in the corresponding proofs for BTP [4] and ETP [7], we define a CSP instance P_o which is consistent if and only if a possible ordering exists. More precisely, this instance has a variable o_i with domain $\{1, \ldots, n\}$ per variable x_i of X. The value of o_i represents the position of the variable x_i in the ordering. We add a constraint involving $\{o_{i_1}, o_{i_2}, \ldots o_{i_k}, o_{i_{k+1}}\}$ and imposing the condition $o_{i_{k+1}} < max(o_{i_1}, o_{i_2}, \ldots o_{i_k})$ for each k+1-tuple of variables $(x_{i_1}, x_{i_2}, \ldots x_{i_k}, x_{i_{k+1}})$ such that each triple of variables $(x_{i_j}, x_{i_{j'}}, x_{i_{k+1}})$ with $1 \leq j \neq j' \leq k$ has at least one broken triangle on $x_{i_{k+1}}$ w.r.t. x_{i_j} and $x_{i_{j'}}$.

If P_o has a solution, then let $<$ be any total ordering of the variables which is a completion of the partial ordering given by the values of the variables o_i. Then for each k+1-tuple of variables $(x_{i_1}, x_{i_2}, \ldots x_{i_k}, x_{i_{k+1}})$, with $i_1 < \ldots < i_{k+1}$, we have at least one triple of variables $(x_{i_j}, x_{i_{j'}}, x_{i_{k+1}})$ with $1 \leq j \neq j' \leq k$ which has no broken triangle on $x_{i_{k+1}}$ w.r.t. x_{i_j} and $x_{i_{j'}}$. Indeed, if this were not the case, then the constraint $o_{i_{k+1}} < max(o_{i_1}, o_{i_2}, \ldots o_{i_k})$ would have been imposed, which is in contradiction with $i_1 < \ldots < i_{k+1}$. So, if P_o has a solution, we have an ordering satisfying the k-BTP property. Conversely, let us consider an ordering satisfying the k-BTP property and assume that P_o has no solution. It means that at least one constraint $o_{i_{k+1}} < max(o_{i_1}, o_{i_2}, \ldots o_{i_k})$ is violated. So each triple of variables $(x_{i_j}, x_{i_{j'}}, x_{i_{k+1}})$ with $1 \leq j \neq j' \leq k$ has at least one broken triangle on $x_{i_{k+1}}$, which is impossible since this ordering satisfies the k-BTP property. Hence P_o has a solution if and only if P admits an ordering satisfying the k-BTP property.

We now prove that P_o can be built and solved in polynomial time. Finding all the broken triples can be achieved in $O(n^3.d^4)$ time, while defining the constraints $o_{i_{k+1}} < max(o_{i_1}, o_{i_2}, \ldots o_{i_k})$ can be performed in $O(n^{k+1})$. So P_o can be computed in $O(n^3.d^4 + n^{k+1})$. Moreover, P_o can be solved in polynomial time by establishing generalized arc-consistency since its constraints are *max-closed* [10]. □

We analyze now the complexity of solving instances of the class k-BTP ($k \geq 3$). The following theorem shows that this is NP-hard since this is true even for the smaller class $ETP \subset 3$-BTP.

Theorem 3 *Deciding whether an instance of the class ETP is satisfiable is NP-complete.*

Proof: It sufficies to exhibit a polynomial reduction from binary CSP to its subproblem ETP. Given any binary CSP instance I, we can construct an equivalent instance I' by

1. adding a new variable x_{ij} (with domain $D_{x_{ij}} = D_{x_i}$) for each constraint c_{ij}
2. adding a new equality constraint between each x_i and x_{ij}
3. replacing each constraint $(\{x_i, x_j\}, R)$ by the constraint $(\{x_{ij}, x_j\}, R)$.

Let $<$ be any variable order in I' in which all the new variables x_{ij} occur after all the original variables x_k. Since each variable is constrained by at most two variables which precede it in this order, we can easily deduce that I' satisfies ETP. It follows from this polynomial reduction that deciding whether an instance of the class ETP is satisfiable

is NP-complete. □

To ensure the tractability of the class k-BTP, we consider an additional condition which is that instances satisfy *Strong k-Consistency* [8].

Definition (Strong k-Consistency) *A binary CSP instance P satisfies i-Consistency if any consistent assignment to $i-1$ variables can be extended to a consistent assignment on any i^{th} variable. A binary CSP instance P satisfies Strong k-Consistency if it satisfies i-Consistency for all i such that $1 < i \leq k$.*

Strong k-Consistency and k-BTP allow us to define a new tractable class:

Theorem 4 *Let P be a binary CSP instance P such that there exists a constant k with $2 \leq k < n$ for which P satisfies both Strong k-Consistency and k-BTP w.r.t. the variable ordering $<$. Then P is consistent and a solution can be found in polynomial time.*

Proof: We consider an ordering for variable assignments corresponding to the ordering $<$. As the instance satisfies Strong k-Consistency, it satisfies arc-consistency and thus, no domain is empty and each value has a support in each other domain. Moreover, as the instance satisfies Strong k-Consistency, we have a consistent assignment on the k first variables. Now, and more generally, suppose that we have a consistent assignment $(u_1, u_2, \ldots u_{l-1}, u_l)$ for the l first variables $x_1, x_2, \ldots x_{l-1}, x_l$ in the ordering, with $k \leq l < n$. We show that this assignment can be consistently extended to the variable x_{l+1}. To show this, we must prove that $\cap_{1 \leq i \leq l} R(c_{il+1})[u_i] \neq \emptyset$, that is there is at least one value in the domain of x_{l+1} which is compatible with the assignment $(u_1, u_2, \ldots u_{l-1}, u_l)$.

We first prove this for $l = k$. Consider the consistent assignment $(u_1, u_2, \ldots u_{k-1}$, $u_k)$ on the k first variables. Consider a $k+1^{th}$ variable x_{k+1} appearing later in the ordering. Since P satisfies k-BTP, there exists at least one triple of variables $(x_j, x_{j'}, x_{k+1})$ with $1 \leq j \neq j' \leq k$ such that there is no broken triangle on x_{k+1} w.r.t. x_j and $x_{j'}$. By Lemma 2.4 given in [4], we have:

$$(R(c_{jk+1})[u_j] \subseteq R(c_{j'k+1})[u_{j'}])$$

or

$$(R(c_{j'k+1})[u_{j'}] \subseteq R(c_{jk+1})[u_j])$$

Without loss of generality, assume that we have $R(c_{jk+1})[u_j] \subseteq R(c_{j'k+1})[u_{j'}]$ and $j < j'$. Since P satisfies Strong k-Consistency, we know that the sub-assignment of $(u_1, u_2, \ldots, u_j, \ldots u_{k-1}, u_k)$ on $k-1$ variables excluding the assignment $u_{j'}$ for $x_{j'}$ can be consistently extended to x_{k+1}. Moreover, we know that $R(c_{jk+1})[u_j] \subseteq R(c_{j'k+1})[u_{j'}]$ and by arc-consistency, $R(c_{i_j i_{k+1}})[u_j] \neq \emptyset$. Thus, $(u_1, u_2, \ldots, u_j, \ldots,$ $u_{j'}, \ldots, u_k, u_{k+1})$ is a consistent assignment to the $k+1$ first variables.

Note that this proof holds for all subsets of $k+1$ variables such that x_{k+1} appears later in the ordering $<$, not only for the $k+1$ first variables $x_1, x_2, \ldots x_{k-1}, x_k$ and x_{k+1}.

Now, we prove the property for l with $k < l < n$. That is, we show that a consistent assignment $(u_1, u_2, \dots u_{l-1}, u_l)$ can be extended to a $(l+1)^{th}$ variable. As induction hypothesis, we assume that every consistent assignment on $l-1$ variables can be extended to a l^{th} variable, which appears later in the considered ordering $<$.

Consider a consistent assignment $(u_1, u_2, \dots u_{l-1}, u_l)$ on the l first variables. Let $(u_{i_1}, u_{i_2}, \dots u_{i_k})$ be a sub-assignment on k variables of the assignment $(u_1, u_2, \dots u_{l-1}, u_l)$. As P satisfies k-BTP, and as $k < l < n$, for all subsets of k variables $x_{i_1}, x_{i_2}, \dots x_{i_k}$, we know that there is a triangle which is not broken in x_{l+1} w.r.t. x_{i_j} and $x_{i_{j'}}$, with x_{i_j} and $x_{i_{j'}}$ appearing in the variables $x_{i_1}, x_{i_2}, \dots x_{i_k}$. So, without loss of generality, we can consider that $i_1 \leq i_j < i_{j'} \leq i_k \leq l$ and we have $R(c_{i_j l+1})[u_{i_j}] \subseteq R(c_{i_{j'} l+1})[u_{i_{j'}}]$. Note that x_{i_j} and $x_{i_{j'}}$ can be interchanged in the ordering if necessary.

Now, consider the consistent assignment $(u_1, u_2, \dots u_{l-1}, u_l)$ on the l first variables. By the induction hypothesis, each partial assignment of $(u_1, u_2, \dots u_{l-1}, u_l)$ on $l-1$ variables can be extended to a consistent assignment on x_{l+1} with a compatible value u_{l+1}. So, consider the partial assignment on $l-1$ variables where $u_{i_{j'}}$ does not appear. This assignment is for example $(u_1, u_2, \dots u_{i_j}, \dots u_{l-1}, u_l, u_{l+1})$. As we have $R(c_{i_j l+1})[u_{i_j}] \subseteq R(c_{i_{j'} l+1})[u_{i_{j'}}]$, the value $u_{i_{j'}}$ is also compatible with u_{l+1}, and thus the assignment $(u_1, u_2, \dots u_{i_j}, \dots u_{i_{j'}}, \dots u_{l-1}, u_l, u_{l+1})$ on the $l+1$ first variables is a consistent assignment.

So, every consistent assignment $(u_1, u_2, \dots u_{l-1}, u_l)$ on $(x_1, x_2, \dots x_{l-1}, x_l)$ can be extended to a $(l+1)^{th}$ variable, for all l with $k < l < n$. And more generally, we have shown that every consistent assignment on l variables, not necessarily consecutive in the ordering (as are the l first variables), can be extended to a consistent assignment for every $(l+1)^{th}$ variable which appears after these l variables in the ordering $<$ associated with k-BTP. Thus, the induction hypothesis holds for the next step.

Note that this proof also shows that an instance which satisfies Strong k-Consistency and k-BTP (with respect to the variable ordering $<$) is consistent.

Finally, given the ordering $<$, we show that finding a solution can be performed in polynomial time. Given a consistent assignment $(u_1, u_2, \dots u_l)$ with $l < n$, finding a compatible value u_{l+1} for the next variable x_{l+1} is feasible by searching in its domain whose size is at most d. For each value, we need to verify the constraints connecting the variable x_{l+1} which can be done in $O(e_{l+1})$ if the next variable x_{l+1} has e_{l+1} neighbors in the previous variables. Since $\Sigma_{1 \leq l < n} e_{l+1} = e$, the total cost to find a solution is $O((n+e).d)$. □

In the sequel, we denote k-BTP-SkC, the class of instances satisfying k-BTP and Strong k-Consistency. One of the most interesting properties of the tractable class BTP is the fact that the instances of this class can be solved in polynomial time using classical algorithms (such as MAC or RFL) implemented in most solvers. The next property establishes that a similar result holds for k-BTP-SkC. Indeed, the proof of Theorem 4 allows us to show that algorithms such as BT (Backtracking), MAC and RFL can solve any instance of the class k-BTP-SkC in polynomial time:

Theorem 5 *Given a binary CSP instance $P = (X, D, C)$ and a variable ordering $<$ such that P satisfies k-BTP w.r.t. $<$, and is Strongly-k-Consistent, the algorithms BT, MAC and RFL find a solution of the instance P in polynomial time.*

Proof: As the instance satisfies Strong k-Consistency, BT using the ordering $<$ for the variable assignment can find a consistent assignment on $x_1, x_2, \ldots x_{k-1}$ and x_k. Moreover, given l, with $k < l < n$, it is shown in the proof of Theorem 4 that a consistent assignment $(u_1, u_2, \ldots u_{l-1}, u_l)$ on $x_1, x_2, \ldots x_{l-1}$ and x_l can be extended to a $(l+1)^{th}$ variable, that is on x_{l+1}. To find the assignment of x_{l+1}, we need to look for a compatible value in its domain. This is feasible in $O(e_{l+1}.d)$ assuming that x_{l+1} has e_{l+1} neighbors in the previous variables. So, as for the proof of Theorem 4, finding a solution of P is globally feasible in $O((n+e).d)$. If we consider now algorithms such as MAC or RFL, by the same reasoning, we show that their complexity is bounded by $O(n.(n+e).d^2)$ due to the additional cost of the arc-consistency filtering performed after each variable assignment. □

In Section 5, we discuss the interest of the class k-BTP from a practical viewpoint. In the next section, we study the relationships between k-BTP and some tractable classes.

4 Relationship with some tractable classes

We consider the important tractable class based on the notion of tree-decomposition of graphs [11].

Definition (Tree-Decomposition) *Given a graph $G = (X, C)$, a tree-decomposition of G is a pair (E, T) with $T = (I, F)$ a tree and $E = \{E_i : i \in I\}$ a family of subsets of X, such that each subset (called cluster or bag in Graph Theory) E_i is a node of T and satisfies:*

1. *$\cup_{i \in I} E_i = X$,*
2. *for each edge $\{x, y\} \in C$, there exists $i \in I$ with $\{x, y\} \subseteq E_i$, and*
3. *for all $i, j, k \in I$, if k is in a path from i to j in T, then $E_i \cap E_j \subseteq E_k$.*

The width of a tree-decomposition (E, T) is equal to $max_{i \in I}|E_i| - 1$. The tree-width w of G is the minimal width over all the tree-decompositions of G.

Let k-TW be the class of binary CSPs instances such that their tree-width is less than or equal to a constant k. Recently, M. Vardi asked a question about the relationships between k-TW and ETP or other generalizations of BTP. The next theorems give a first partial answer to this question.

Theorem 6 *k-$TW \subsetneq (k+1)$-BTP.*

Proof: We show firstly that k-$TW \subseteq (k+1)$-BTP. It is well known that if the tree-width of a binary instance of CSP is bounded by k, there is an ordering $<$ on variables, such that for $x_i \in X$, $|\{x_j \in X : j < i \text{ and } c_{ji} \in C\}| \leq k$ [2]. Now, consider a subset

of $k+2$ variables $x_{i_1}, x_{i_2}, \ldots x_{i_k}, x_{i_{k+1}}, x_{i_{k+2}}$ such that $i_1 < i_2 < \ldots < i_{k-1} < i_k < i_{k+1} < i_{k+2}$. Since the tree-width is bounded by k, we know that there are at most k constraints $c_{i_j i_{k+2}} \in C$. So, there is at least one triple of variables $(x_{i_j}, x_{i_{j'}}, x_{i_{k+2}})$ with $1 \leq j \neq j' \leq k$ such that $c_{i_j i_{k+2}} \notin C$ or $c_{i_{j'} i_{k+2}} \notin C$. Without loss of generality, assume that there is no constraint $c_{i_j i_{k+2}} \in C$. Thus, there is no broken triangle on $x_{i_{k+2}}$ w.r.t. x_{i_j} and $x_{i_{j'}}$ because all the values of $D_{x_{i_j}}$ are compatible with all the values of $D_{x_{i_{k+2}}}$. So, the considered instance of CSP satisfies the property $(k+1)$-BTP. Finally, it is easy to define instances whose tree-width is strictly greater than k which satisfy the property $(k+1)$-BTP. For example, we can consider an instance of CSP with domains of size one, with the complete constraint graph, and with one solution. The tree-width of this instance is $n-1$ while it satisfies k-BTP for all possible values of k. □

The cost of checking for satisfiability of instances in k-TW has a similar cost to that of achieving Strong $(k$+1)-Consistency, that is $O(n^{k+1}d^{k+1})$. Nevertheless, this does not allow us to establish a formal inclusion of k-TW in $(k$+1)-BTP-$S(k$+1)C which is tractable while $(k$+1)-BTP is NP-complete for $k \geq 2$ by Theorem 3. But if we denote k-TW-$S(k$+1)C, the class of binary CSPs instances belonging to k-TW and which satisfy Strong $(k$+1)-Consistency, the next result holds:

Theorem 7 k-TW-$S(k$+1)$C \subsetneq (k+1)$-BTP-$S(k$+1)C.

The tractable class BTP has also recently been generalized in a different way to that proposed in this paper, again by noticing that not all broken triangles need to be forbidden [12]. We will show that these two generalizations are orthogonal.

Definition (∀∃-BTP) *A binary CSP instance P satisfies the property ∀∃-BTP w.r.t. the variable ordering $<$ if, and only if, for each pair of variables x_i, x_k such that $i < k$, for all $v_i \in D_{x_i}$, $\exists v_k \in D_{x_k}$ such that $(v_i, v_k) \in R(c_{ik})$ and for all x_j with $j < k$ and $j \neq i$, and for all $v_j \in D_{x_j}$ and for all $v'_k \in D_{x_k}$, (v_i, v_j, v_k, v'_k) is not a broken triangle on x_k w.r.t. x_i and x_j. Let ∀∃-BTP be the set of the instances for which ∀∃-BTP holds w.r.t. some variable ordering.*

The class ∀∃-BTP can be solved and recognized in polynomial time [12]. It represents a tractable class which strictly includes BTP since it does not forbid all broken triangles. Since k-BTP also does not forbid all broken triangles, it is natural to compare these two classes. We do this for the special case $k = 3$, but the same argument applies for any value of $k \geq 3$.

Theorem 8 *Even for sets of binary CSP instances which are strong path consistent, the properties 3-BTP and ∀∃-BTP are incomparable.*

Proof: Consider an instance P^* in which each domain D_{x_k} contains a value a^* such that for all other variables x_i, for all values $v_i \in D_{x_i}$, $(v_i, a^*) \in R(c_{ik})$. Then P^* satisfies ∀∃-BTP since there can be no broken triangle of the form (v_i, v_j, a^*, v'_k), the value a^* being compatible with all assignments to all other variables. It is easy to complete such

an instance P^* so that it does not satisfy 3-BTP for any variable ordering by adding broken triangles on domain elements other than a^*.

Consider a 3-variable binary CSP instance P_3 with domains $\{0, 1, 2\}$ and the following three constraints: $x_1 \neq x_2$, $x_1 \neq x_3$, $x_2 \neq x_3$, i.e. a 3-colouring problem on a complete graph on three vertices. Then P_3 is strong path consistent and trivially satisfies 3-BTP (since there are only 3 variables), but P_3 does not satisfy $\forall\exists$-BTP for any ordering $i < j < k$ of the variables (due to the existence of broken triangles on assignments (x_i, a), (x_j, b), (x_k, a), (x_k, b) for all pairs of distinct colours a, b). □

We now consider a very general tractable class recently discovered by Naanaa [9] and which undoubtedly deserves to be better known.

Let E be a finite set and let $\{E_i\}_{i\in I}$ be a finite family of subsets of E. The family $\{E_i\}_{i\in I}$ is said to be *independent* if and only if for all $J \subset I$,

$$\bigcap_{i\in I} E_i \;\subset\; \bigcap_{j\in J} E_j$$

(where the notation $A \subset B$ means that A is a proper subset of B). Observe that $\{E_i\}_{i\in I}$ cannot be independent if $\exists j \neq j' \in I$ such that $E_j \subseteq E_{j'}$, since in this case and with $J = I \setminus \{j'\}$ we would have

$$\bigcap_{i\in I} E_i \;=\; \bigcap_{j\in J} E_j.$$

Definition (Directional Rank) *Let P be a binary CSP instance whose variables are totally ordered by $<$. The* directional rank *of variable x_m is the size k of the largest consistent assignment $(a_1, \ldots, a_k)$ to a set of variables $x_{i_1}, \ldots, x_{i_k}$ (with $i_1 < \ldots < i_k < m$) such that the family of sets $\{R(c_{i_j m})[a_j]\}_{j=1,\ldots,k}$ is independent. The directional rank of P (w.r.t the ordering $<$ of its variables) is the maximum directional rank over all its variables.*

Naanaa has shown that if P is a binary CSP instance which has directional rank no greater than k and is directional strong $(k+1)$-consistent then I is globally consistent [9]. We denote DR-k, the set of these instances. Naanaa points out that some known tractable classes, such as binary CSP instances with connected row convex constraints [13], have bounded directional rank.

If a binary CSP instance P is $(k+1)$-BTP, then no variable can have a directional rank greater than k. This is because for any variable x_m and any assignments $(a_1, \ldots, a_{k+1})$ to any set of variables $x_{i_1}, \ldots, x_{i_{k+1}}$ with $i_1 < \ldots < i_{k+1} < m$, by the definition of $(k+1)$-BTP, we must have $R(c_{i_j m})[a_j] \subseteq R(c_{i_{j'} m})[a_{j'}]$ for some $j \neq j' \in \{1, \ldots, k+1\}$; hence, as observed above, the sets $\{R(c_{i_j m})[a_j]\}_{j=1,\ldots,k+1}$ cannot be independent. It follows that the tractability of $(k$+1$)$-BTP-$S(k$+1$)C$ is also a corollary of the result of Naanaa [9]. On the other hand, the property $(k+1)$-BTP, although subsumed by DR-k, can be detected in time complexity $O(n^k d^k + n^3 d^4)$ compared to $O(n^{k+1} d^{k+1})$ for DR-k.

5 Experiments

In this section, we compare the tractable classes BTP, $ETP\text{-}SPC$, $k\text{-}BTP\text{-}SkC$ and $DR\text{-}k\text{-}1$ (where SPC stands for Strong Path Consistency) from a practical viewpoint. We only consider the case $k = 3$, since strong k-consistency becomes too expensive in time for $k > 3$ and may add constraints of arity $k - 1$.

Tractable classes are often critized for being artificial in the sense that their underlying properties seldom occur in real instances. So, here, we first highlight the existence of instances belonging to some of these classes among the benchmark instances classically exploited for solver evaluations and comparisons. More precisely, our experiments involve 2,373 binary benchmarks from the third CSP Solver Competition[3] and cover all the benchmarks exploited in [7].

Then we will investigate the possible link between efficient solving and belonging to these tractable classes.

5.1 Instances belonging to tractable classes

Since the tractable classes $ETP\text{-}SPC$, 3-$BTP\text{-}SPC$ and DR-2 require strong path-consistency, we first achieve SPC on each instance before checking whether it belongs to the considered classes, in the same spirit as [14, 15]. In so doing, 628 instances are detected as inconsistent and so they trivially belong to all of these tractable classes. 85 of the remaining instances belong to 3-$BTP\text{-}SPC$ while 87 have directional rank at most two. Among these instances, we have respectively 71 and 76 instances in $BTP\text{-}SPC$ and $ETP\text{-}SPC$. These differences between these tractable classes are well highlighted by some instances of the bqwh-15-106 family since we can observe all the possible configurations of the relations $BTP\text{-}SPC \subsetneq ETP\text{-}SPC \subsetneq$ 3-$BTP\text{-}SPC \subsetneq DR$-2. For example, instance bqwh-15-106-13 belongs to all the considered tractable classes while instances bqwh-15-106-28, bqwh-15-106-16 and bqwh-15-106-76 only belong respectively to three, two or one of these tractable classes. Table 1 presents some instances belonging to classes $ETP\text{-}SPC$, 3-$BTP\text{-}SPC$ or DR-2. It also provides the tree-width w of these instances and their tree-width w' once SPC is enforced. When the exact tree-width is unknown (recall that computing an optimal tree-decomposition is an NP-hard problem), we give a range. We can note the diversity of these instances (academic, random or real-world instances). Some of these instances belong to 3-$BTP\text{-}SPC$ or DR-2 thanks to their structure. For instance, graph12-w0 and hanoi-7 have an acyclic constraint graph while the tree-width of domino-100-100 and crossword-m1-uk-puzzle01 is two. However, most instances have a tree-width greater than two. Moreover, in most cases, the application of SPC may significantly increase the original tree-width of these instances. For example, the tree-width of instance driverlogw-09-sat is initially bounded by 108 and is equal to 629 after the application of SPC. This increase is explained by the pairs of values which are forbidden by SPC. When SPC forbids a pair of values (v_i, v_j) for a given pair of variables (x_i, x_j), it removes (v_i, v_j) from the relation $R(c_{ij})$ if the constraint c_{ij} exists. However, if the constraint c_{ij} does not exist yet, SPC must first add it to the problem. In such a case, depending on the added constraints

[3] See http://www.cril.univ-artois.fr/CPAI08.

Table 1. Some instances belonging to $BTP\text{-}SPC$, $ETP\text{-}SPC$, 3-$BTP\text{-}SPC$ or DR-2 after the application of SPC with their tree-width w and the tree-width w' of the instances once SPC is enforced.

Instance	n	w	w'	$BTP\text{-}SPC$	$ETP\text{-}SPC$	3-$BTP\text{-}SPC$	DR-2
bqwh-15-106-13	106	$[7, 48]$	104	yes	yes	yes	yes
bqwh-15-106-16	106	$[6, 45]$	99	no	no	yes	yes
bqwh-15-106-28	106	$[7, 52]$	105	no	yes	yes	yes
bqwh-15-106-76	106	$[6, 44]$	100	no	no	no	yes
bqwh-15-106-77	106	$[7, 50]$	100	no	no	yes	yes
bqwh-18-141-33	141	$[7, 64]$	134	yes	yes	yes	yes
bqwh-18-141-57	141	$[7, 66]$	137	yes	yes	yes	yes
domino-100-100	100	2	2	yes	yes	yes	yes
domino-5000-500	5000	2	2	yes	yes	yes	yes
driverlogw-04c-sat	272	$[19, 56]$	$[214, 221]$	no	no	no	yes
driverlogw-09-sat	650	$[39, 108]$	629	yes	yes	yes	yes
fapp17-0300-10	300	$[6, 153]$	$[6, 154]$	yes	yes	yes	yes
fapp18-0350-10	350	$[5, 192]$	$[12, 199]$	yes	yes	yes	yes
fapp23-1800-9	1800	$[6, 1325]$	$[41, 1341]$	yes	yes	yes	yes
graph12-w0	680	1	1	yes	yes	yes	yes
graph13-w0	916	1	1	yes	yes	yes	yes
hanoi-7	126	1	1	yes	yes	yes	yes
langford-2-4	8	7	7	yes	yes	yes	yes
lard-83-83	83	82	82	no	no	yes	yes
lard-91-91	91	90	90	no	no	yes	yes
os-taillard-4-100-0	16	$[3, 9]$	15	yes	yes	yes	yes
os-taillard-4-100-9	16	$[3, 9]$	15	yes	yes	yes	yes
scen5	400	$[11, 32]$	$[167, 188]$	no	no	yes	yes

and their number, the tree-width may significantly increase. Note that the considered instances whose tree-width is initially at most two have a tree-width unchanged by the application of SPC.

5.2 Link between efficient solving and belonging to tractable classes

In this subsection, our aim is not to provide a new module based on tractable classes in order to improve the efficiency of solvers but to see whether we can exploit some tractable classes to explain the efficiency of solvers on some instances. Indeed, we think that tractable classes are more useful from a practical viewpoint if they are implicitly handled by classical solvers than by ad-hoc methods (as is generally the case). For instance, it is well kwown that MAC can solve in backtrack-free manner any binary CSP whose constraint network is acyclic without knowing that the instance has this particular feature [16].

Most state-of-the-art solvers rely on variants of MAC or RFL algorithms. In the following, we focus our study on MAC but we have observed similar results for RFL.

As far as solving is concerned, all the instances belonging to 3-$BTP\text{-}SPC$ or DR-2 are solved in a backtrack-free manner by MAC except the instance driverlogw-04c-sat

which needs one backtrack. Note that MAC has no knowledge about the variable ordering used to satisfy 3-BTP or to obtain a directional rank of at most two. In most cases, we have observed that the ordering CSP instance built in the proof of Theorem 2 in order to compute a suitable variable ordering has no constraints. So any variable ordering is suitable. In contrast, for about a dozen instances, this CSP has several constraints but remains clearly under-constrained and the constraint network has several connected components. This ensues that the ordering CSP in general a huge number of solutions. So it is very likely that MAC exploits implicitly one of these suitable variable orderings. For example, the ordering CSP for checking whether the bqwh-15-106-76 instance (which has 106 variables) has a directional rank at most two has 65 connected components and admits more than 33 million solutions.

Some of the instances are solved efficiently by MAC in a backtrack-free manner even though they do not belong to one of the studied tractable classes. Hence, we now consider the notion of backdoor [17] with the aim in view to provide some explanation about this efficiency in the same spirit as [7]. A *backdoor* is a set of variables defined with respect to a class such that once the backdoor variables are assigned, the problem falls in the class. Here, we are interested in backdoors which are discovered implicitly by MAC when it assigns some variables. Indeed, after some assignments and the associated filtering, the remaining part of the problem may become tractable. So we assess the number of variables which must be assigned before MAC finds implicitly a backdoor w.r.t. one of the studied classes. In practice, over the 50 considered instances, we observe that MAC finds a backdoor w.r.t. BTP after having assigned more variables than for the other considered classes. The numbers of assigned variables required to find a backdoor respectively for ETP and 3-BTP are very close, and even equal in most cases. By considering DR-2, we save a few variables compared to ETP and 3-BTP. For example, MAC needs to assign at most five variables before finding a backdoor w.r.t. to 3-BTP or DR-2 for 14 instances compared to 12 and 4 instances, respectively, for ETP and BTP[4]. Of course, the resulting instances do not necessarily satisfy strong path-consistency and so we cannot exploit directly Theorem 5 to explain the efficiency of MAC. Nevertheless, when the instance is 3-BTP and strong path-consistent after having assigned some variables, MAC may exploit implicitly a suitable variable ordering since, as evoked above, the corresponding ordering CSP often admits a large number of solutions. Furthermore Theorem 5 provides sufficient conditions so that MAC solves some instances in polynomial time, but these conditions are not always necessary. For instance, MAC solves the instances which belong to BTP in polynomial time without requiring a suitable variable ordering or the satisfaction of strong path-consistency. Hence, one part of the explanation of the practical efficiency of MAC may lie in its ability to exploit implicitly different tractable classes.

[4] Note that these instances do not include all the instances mentioned in [7] since some of them belong to 3-BTP-SPC and/or DR-2.

6 Conclusion

This paper introduces a novel family of tractable classes for binary CSPs, denoted k-BTP whose tractability is associated with a given level of strong k-consistency. It is based on a hierarchy of classes of instances with the BTP class as the base case. While BTP is defined on subsets of 3 variables, the k-BTP class is defined on sets of k+1 variables, while relaxing the restrictive conditions imposed by BTP which is the class 2-BTP. We showed that k-BTP inherits some of the desirable properties of BTP, such as polynomial solvability using standard algorithms such as MAC. We also showed that k-BTP strictly generalizes the class of instances whose tree-width is bounded by a constant and we analyzed the relationships with the class based on the notion of *directional rank* recently introduced by Naanaa. To assess the practical interest of the k-BTP class, an experimental analysis is presented focusing on the particular case of 3-BTP. This analysis shows a significant advantage of 3-BTP compared to BTP and to CSPs of bounded tree-width.

Further research is required to determine if the condition corresponding to strong k-consistency is actually necessary or whether a weaker condition would suffice. Indeed, experiments showed that MAC can solve without backtracking certain instances belonging to 3-BTP even when they do not verify the corresponding level of consistency. From a practical point of view, an interesting challenge is to find the minimum (generally) required level of consistency among different kinds of local consistencies such as PIC [18], maxRPC [19] or SAC [20]. Note that, from a theoretical point of view, we can easily deduce from Theorem 3 that any local consistency that only performs domain filtering (e.g. PIC, maxRPC, SAC) cannot be sufficient (assuming P$\neq$NP) since ETP is invariant under domain filtering operations.

Moreover, studying a relaxation of the k-BTP condition needs to be addressed so as to further expand the class of instances that can be solved in polynomial time, but along different avenues to the one proposed in [9], even if further theoretical and experimental research are clearly required to fully appreciate all the consequences of Naanaa's result. Finally, it could be interesting to investigate a similar approach to the one introduced in [21] which provides a novel polynomial-time reduction operation based on the merging of domain values.

References

1. F. Rossi, P. van Beek, and T. Walsh. *Handbook of Constraint Programming*. Elsevier, 2006.
2. R. Dechter and J. Pearl. Tree-Clustering for Constraint Networks. *Artificial Intelligence*, 38:353–366, 1989.
3. G. Gottlob, N. Leone, and F. Scarcello. A Comparison of Structural CSP Decomposition Methods. *Artificial Intelligence*, 124:343–282, 2000.
4. M.C. Cooper, P. Jeavons, and A. Salamon. Generalizing constraint satisfaction on trees: hybrid tractability and variable elimination. *Artificial Intelligence*, 174:570–584, 2010.
5. D. Sabin and E. Freuder. Contradicting Conventional Wisdom in Constraint Satisfaction. In *Proceedings of ECAI*, pages 125–129, 1994.
6. B. Nadel. *Tree Search and Arc Consistency in Constraint-Satisfaction Algorithms*, pages 287–342. In *Search in Artificial Intelligence*. Springer-Verlag, 1988.

7. P. Jégou and C. Terrioux. The Extendable-Triple Property: a new CSP Tractable Class beyond BTP. In *Proceedings of AAAI*, pages 3746–3754, 2015.
8. E. Freuder. A Sufficient Condition for Backtrack-Free Search. *Journal of the ACM*, 29 (1):24–32, 1982.
9. W. Naanaa. Unifying and extending hybrid tractable classes of csps. *Journal of Experimental and Theoretical Artificial Intelligence*, 25(4):407–424, 2013.
10. P. Jeavons and M. Cooper. Tractable constraints on ordered domains. *Artificial Intelligence*, 79(2):327–339, 1995.
11. N. Robertson and P.D. Seymour. Graph minors II: Algorithmic aspects of treewidth. *Algorithms*, 7:309–322, 1986.
12. M.C. Cooper. Beyond consistency and substitutability. In *Proceedings of CP*, pages 256–271, 2014.
13. Y. Deville, O. Barette, and P. van Hentenryck. Constraint satisfaction over connected row convex constraints. *Artificial Intelligence*, 109(1-2):243–271, 1999.
14. A. El Mouelhi, P. Jégou, and C. Terrioux. Hidden Tractable Classes: from Theory to Practice. In *Proceedings of ICTAI*, pages 437–445, 2014.
15. A. El Mouelhi, P. Jégou, and C. Terrioux. Hidden Tractable Classes: from Theory to Practice. *Constraints*, 2015.
16. D. Sabin and E. Freuder. Understanding and Improving the MAC Algorithm. In *Proceedings of CP*, pages 167–181, 1997.
17. R. Williams, C.P. Gomes, and B. Selman. Backdoors to typical case complexity. In *Proceedings of IJCAI*, pages 1173–1178, 2003.
18. E. Freuder and C.D. Elfe. Neighborhood inverse consistency preprocessing. In *Proceedings of AAAI*, pages 202–208, 1996.
19. R. Debruyne and C. Bessière. From restricted path consistency to max-restricted path consistency. In *Proceedings of CP*, pages 312–326, 1997.
20. R. Debruyne and C. Bessière. Domain Filtering Consistencies. *Journal of Artificial Intelligence Research*, 14:205–230, 2001.
21. M.C. Cooper, A. El Mouelhi, C. Terrioux, and B. Zanuttini. On broken triangles. In *Proceedings of CP*, pages 9–24, 2014.