Constraints manuscript No.
(will be inserted by the editor)

On a new extension of BTP for binary CSPs

Achref El Mouelhi

Received: date / Accepted: date

Abstract The study of broken-triangles is becoming increasingly ambitious, by both solving constraint satisfaction problems (CSPs) in polynomial time and reducing search space size through either value merging or variable elimination. Considerable progress has been made in extending this important concept, such as dual broken-triangle and weakly broken-triangle, in order to maximize the number of captured tractable CSP instances and/or the number of merged values. Specifically, m-wBTP allows us to merge more values than BTP. DBTP, $\forall\exists$-BTP, k-BTP, WBTP and m-wBTP permit us to capture more tractable instances than BTP. However, except BTP, none of these extensions allows variable elimination while preserving satisfiability. Moreover, k-BTP and m-wBTP define bigger tractable classes around BTP but both of them generally need a high level of consistency.

Here, we introduce a new weaker form of BTP, called m-fBTP for flexible broken-triangle property, which will represent a compromise between most of these previous tractable properties based on BTP. m-fBTP allows us on the one hand to eliminate more variables than BTP while preserving satisfiability and on the other to define a new bigger tractable class for which arc consistency is a decision procedure. Likewise, m-fBTP permits to merge more values than BTP but fewer than m-wBTP. The binary CSP instances satisfying m-fBTP are solved by algorithms of the state-of-the-art like MAC and RFL in polynomial time. An open question is whether it is possible to compute, in polynomial time, the existence of some variable ordering for which a given instance satisfies 1-fBTP.

1 Introduction

A wide range of real-life problems issue from Artificial Intelligence (AI) and Operational Research (OR) like spatial and temporal planning, scheduling and configuration can be expressed as a binary *Constraint Satisfaction Problems* (CSPs [1]). A binary CSP consists of a set of *variables* X, each one has a finite set of *values* called

A. El Mouelhi
Marseille 13015, France
E-mail: {elmouelhi.achref}@gmail.org

domain D, and a finite set of *constraints* C. Each constraint is defined over a pair of variables and represents a set of valid assignment of values to these two variables, involved by the constraint. A *solution* to a CSP instance is an assignment of value to each variable satisfying all the constraints. Checking whether a given CSP instance has a solution is known to be NP-complete.

In general, the main techniques to achieve this task are based on backtracking algorithms, whose worst-case time complexity is $O(ed^n)$ where e, n and d are the number of constraints, the number of variables and the maximum domain size, respectively. In order to reduce this exponential time complexity, many different approaches have been proposed. The first one, called *filtering by consistency*, consists of removing inconsistent values [2], values which cannot take part in any solution. Obviously this approach leaves the set of solutions unchanged. The second relies on *merging values* (consistent or inconsistent), satisfying some conditions, without affecting the existence of a solution [3,4,5,6]. The last *eliminates* variables [7,8] or constraints [9] while preserving the satisfiability of the instance.

In a somewhat orthogonal direction, much research has been devoted to identifying *tractable classes*. In the literature, several tractable classes have been defined but the *Broken-Triangle Property* (BTP [10,11]) still remains at the heart of this research area. This property has some interesting characteristics from a solving viewpoint as well as reduction operations viewpoint. Indeed, BTP is not only defined for solving CSP in polynomial time, but also for reducing the size of CSP instances while preserving satisfiability. Specifically, the absence of broken-triangles has led, under some conditions, to variable elimination [7,8] or domain reduction by value merging [5,6] while preserving satisfiability.

More recently, it has been proved that the presence of certain broken-triangles does not necessarily preclude defining tractable classes [12,13,14,15,16,17,18,19] and/or merging values [19]. For example, ETP [16] and more generally k-BTP [17] authorises some broken-triangles and defines larger tractable classes than BTP but does not permit value merging. Likewise, m-wBTP [19] does not forbid all broken-triangles and defines a maximal value-merging condition. Unfortunately, none of them allow variable elimination (see [19]) although the initial definition of BTP permits it [8]. Moreover, k-BTP seems to be unusable beyond $k = 3$ and m-wBTP appears to be inexploitable when $m > 2$ because of the level of consistency required.

The main contribution of this work is providing a new weaker-form of BTP, called m-fBTP, which allows value merging, variable elimination and defines a new hybrid tractable class for which arc consistency is a decision procedure. The results proven in this paper also provide theoretical insight into the relationship between m-fBTP and some others previous extension of BTP.

So, our paper will be organised as follows: Section 2 recalls some definitions and notations. In section 3, we introduce the flexible broken-triangle property. Next, we show that m-fBTP is a maximal variable-elimination condition. Section 5 proves that m-fBTP instances defines a tractable class which can be efficiently solved by algorithms of the state-of-the-art like MAC [20] and RFL [21]. In section 6, we compare m-fBTP to some known tractable classes like DBTP, $\forall\exists$-BTP [13], k-BTP, m-wBTP and WBTP [18]. After, we experimmentally show the existence of our patterns in benchmark problems of the CSP competition 2008. Finally we give a discussion and perspective for future work. Some results of this paper first published in [22].

2 Formal background

Constraint satisfaction problems constitute an important tool for modeling and solving many different practical problems in Artificial Intelligence and Operations Research. A non-binary CSP instance, also called n-ary, is defined as below:

Definition 1 (CSP instance) A CSP instance is a pair $I = (X, C)$ with:

- X: a set of n **variables** denoted by $\{x_1, ..., x_n\}$. Each variable x_i has a **domain** $D(x_i)$ containing at most d values.
- C: a set of e **constraints**. Each constraint C_i is a pair $(Scp(C_i), Rel(C_i))$ where:
  - $Scp(C_i) = \{x_{i_1}, ..., x_{i_{a_i}}\} \subseteq X$, is the **scope** of C_i,
  - $Rel(C_i) \subseteq D(x_{i_1}) \times ... \times D(x_{i_{a_i}})$, is the associated **relation**.

Recall that any non-binary CSP instance can be converted into an equivalent binary instance by using dual encoding or hidden-variable transformation[23,24]. In this paper, we consider only binary CSP instances, defined formally as follows:

Definition 2 (Binary CSP instance) A binary CSP instance is a pair $I = (X, C)$ with:

- X: a set of n **variables** denoted by $\{x_1, ..., x_n\}$. Each variable x_i has a **domain** $D(x_i)$ containing at most d values.
- C: a set of e binary **constraints**. Each binary constraint C_{ij} (with $i \neq j$) is a pair $(Scp(C_{ij}), Rel(C_{ij}))$ where:
  - $Scp(C_{ij}) = \{x_i, x_j\} \subseteq X$, is the **scope** of C_{ij}, i.e. a set of two variables involved by the constraint.
  - $Rel(C_{ij}) \subseteq D(x_i) \times D(x_j)$, is the associated **relation**, a set of compatible pair of values called **tuples**.

If the constraint C_{ij} is not defined in C, then we consider C_{ij} to be a universal constraint (i.e. such that $Rel(C_{ij}) = D(x_i) \times D(x_j)$).

Given a binary CSP instance $I = (X, C)$, an *assignment* of values to $Y \subseteq X$ is a set of pairs $\{(x_i, v_i) \mid x_i \in Y\}$ with $v_i \in D(x_i)$ and $1 \leq i \leq n$, denoted generally $(v_1, ...v_k)$. A *partial solution* of $Y = \{x_{\ell_1}, \ldots, x_{\ell_m}\}$ is an assignment $\mathcal{A} = (v_{\ell_1}, \ldots, v_{\ell_m}) \in D(x_{\ell_1}) \times \ldots \times D(x_{\ell_m})$ which satisfies all constraints C_{ij} such that $\{x_i, x_j\} \subseteq Y$. A partial solution $\mathcal{A}$ is said to be *complete solution* (or *solution* for short) if $Y = X$ i.e. if

- $\mathcal{A}$ contains a value to each variable in X
- no pair of values of $\mathcal{A}$ violates any constraint of C

Given a CSP instance I, deciding whether I has a solution is well known to be NP-complete even for binary CSPs. Neverthless, there are some cases for which solving can be realized in polynomial time. In this case we speak about *tractable* classes. For example, BTP (for *Broken-Triangle Property* [11]) represents an important tractable class from a solving viewpoint as well as reduction operations viewpoint. BTP requires the *absence* of broken-triangles with respect to a given variable ordering. Formally, it is defined as follows:

Definition 3 (Broken-Triangle Property [10,11]) Given a binary CSP instance I with a variable order $<$. A pair of values $v'_k, v''_k \in D(x_k)$ satisfies **BTP** if, for each pair of variables (x_i, x_j) (with $i \neq j \neq k$) such that, $\forall v_i \in D(x_i)$, $\forall v_j \in D(x_j)$, if

- $(v_i, v_j) \in Rel(C_{ij})$,
- $(v_i, v'_k) \in Rel(C_{ik})$ and
- $(v_j, v''_k) \in Rel(C_{jk})$,

then

- either $(v_i, v''_k) \in Rel(C_{ik})$
- or $(v_j, v'_k) \in Rel(C_{jk})$.

A variable x_k satisfies BTP if each pair of values in $D(x_k)$ satisfies BTP. The instance I satisfies BTP with respect to $<$ if for all variables x_k, x_k satisfies BTP in the sub-instance of I on variables $x_i \leq x_k$.

If $(v_i, v''_k) \notin Rel(C_{ik})$ and $(v_j, v'_k) \notin Rel(C_{jk})$, we say that (v'_k, v_i, v_j, v''_k) constitute a *broken-triangle* on the values v'_k and v''_k (or more generally on x_k).

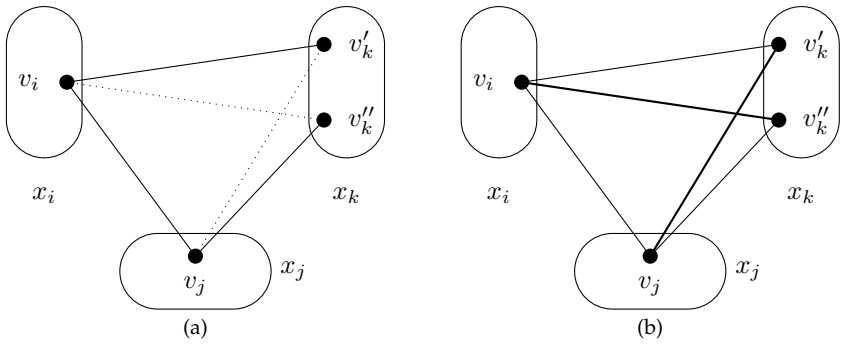

Fig. 1 The assignments (v'_k, v_i, v_j, v''_k) form a broken-triangle in (a) but do not in (b).

Graphically, BTP can be represented in the micro-structure[1] [25] of I as shown in Figure 1. For each pair of values (v_i, v_j) (with $v_i \in D(x_i)$, $v_j \in D(x_j)$ and $i \neq j$), a solid line will be used to connect v_i and v_j if they are compatible (i.e. $(v_i, v_j) \in Rel(C_{ij})$), a dotted line if they are incompatible (i.e. $(v_i, v_j) \notin R_{ij}$), or no line if (v_i, v_j) is an undefined tuple (i.e. a tuple which can be valid or invalid).

Considering the variable ordering $x_i < x_j < x_k$, the CSP instance of Figure 1(a) is not BTP because of the incompatibility of (v_j, v'_k) and (v_i, v''_k). Thus, (v'_k, v_i, v_j, v''_k) constitute a broken-triangle. In Figure 1(b), $(v_i, v''_k) \in Rel(C_{ik})$ and $(v_j, v'_k) \in Rel(C_{jk})$, then the BTP is satisfied. The set of binary CSP instances which satisfy BTP constitutes a tractable class solved by enforcing Arc Consistency.

Definition 4 [2] Given a binary CSP instance $I = (X, C)$, a value $v_i \in D(x_i)$ is arc-consistent with respect to $C_{ij} \in C$ if and only if there exists a value $v_j \in D(x_j)$ such that $(v_i, v_j) \in Rel(C_{ij})$. A domain $D(x_i)$ is arc-consistent with respect to $Rel(C_{ij})$ if and only of $\forall v_i \in D(x_i)$, the value v_i is arc-consistent with respect to $Rel(C_{ij})$, and

[1] Given a binary CSP instance $I = (X, C)$, the *micro-structure* of I is the undirected graph $\mu(I) = (V, E)$ with:

- $V = \{(x_i, v_i) : x_i \in X, v_i \in D(x_i)\}$,
- $E = \{ \{(x_i, v_i), (x_j, v_j)\} \mid i \neq j, C_{ij} \notin C$ or $C_{ij} \in C, (v_i, v_j) \in Rel(C_{ij})\}$

the binary CSP instance I is arc-consistent if and only if $\forall x_i \in X$, the domain $D(x_i)$ is arc-consistent with respect to all $Rel(C_{ij})$ such that $C_{ij} \in C$.

Enforcing arc consistency consists of removing any value that is not arc-consistent. After enforcing arc consistency, if no domain has been wipped out, the binary CSP instance is consistent otherwise it is inconsistent. We have to mention that enforcing arc consistency preserves equivalence (and also satisfiability)

We now define value merging and variable elimination operations.

Definition 5 [5,6] **Merging** the values $v'_k, v''_k \in D(x_k)$ in a binary CSP instance I consists of replacing $v'_k, v''_k \in D(x_k)$ by a new value v_k which is compatible with all values which are compatible with either v'_k or v''_k. A **value-merging condition** is a polytime-computable property such that when it holds on a pair of values $v'_k, v''_k \in D(x_k)$, the instance obtained after merging the values v'_k and v''_k is satisfiable if and only if I was satisfiable.

Definition 6 [7,8] **Eliminating** a variable x_k in a binary CSP instance $I = (X, C)$ consists of replacing X by $X \setminus \{x_k\}$ and C by $C \setminus \{C_{ik} \in C \mid i \neq k\}$. A **variable-elimination condition** is a polytime-computable property such that when it holds on a variable x_k, the instance obtained after eliminating x_k is satisfiable if and only if I was satisfiable.

In [8], it has been shown that if there is no broken-triangle on each pair of values of a given variable x_k in an arc-consistent binary CSP instance I, then x_k can be eliminated from I without changing the satisfiability. For example, the variable x_k of Figure 1(b) can be eliminated while preserving satisfiability, contrary to x_k of Figure 1(a).

In [6], the authors have proved that even when this rule cannot be applied because of the presence of some broken-triangles, it is possible that there is a pair of values v'_k, v''_k in $D(x_k)$ which satisfies BTP. In this case, these two values are mergeable. For example, in Figure 1(b), the values v'_k and v''_k are mergeable.

More recently, [19] showed that even when some broken-triangles are present on a pair of values v'_k, v''_k which satisfies m-wBTP, merging v'_k and v''_k does not affect the satisfiability. Formally, m-wBTP is defined as follows:

Definition 7 (Weakly Broken-Triangle Property [19]) A pair of values $v'_k, v''_k \in D(x_k)$ satisfies m**-wBTP** where $m \leq n - 3$ if for each broken-triangle (v'_k, v_i, v_j, v''_k) with $v_i \in D(x_i)$ and $v_j \in D(x_j)$, there is a set of $r \leq m$ **support variables** $\{x_{\ell_1}, \ldots, x_{\ell_r}\} \subseteq X \setminus \{x_i, x_j, x_k\}$ such that for all $(v_{\ell_1}, \ldots, v_{\ell_r}) \in D(x_{\ell_1}) \times \ldots \times D(x_{\ell_r})$, if $(v_{\ell_1}, \ldots, v_{\ell_r}, v_i, v_j)$ is a partial solution, then there is $\alpha \in \{1, \ldots, r\}$ such that $(v_{\ell_\alpha}, v'_k), (v_{\ell_\alpha}, v''_k) \notin Rel(C_{\ell_\alpha k})$.

Graphically, this definition can be represented through the micro-structure graph of Figure 2. The pair v'_k, v''_k satisfies 1-wBTP because the value v_ℓ in $D(x_\ell)$ is compatible with both v_i and v_j but is not with v'_k and v''_k. So we say that the assignments (v'_k, v_i, v_j, v''_k) forms a *weakly* broken-triangle which is supported by x_ℓ.

m-wBTP was designed primarily to merge a maximum number of values. This property seems to be very weak and it may be for this reason that it does not allow variable elimination (contrary to BTP) and it requires a high level of filtering by consistency to be polynomial. So, next section introduces the *flexible broken-triangle* concept which allows merging value and variable elimination while preserving satisfiability. Like m-wBTP, this new property will use the support variable concept but in a more restrictive way.

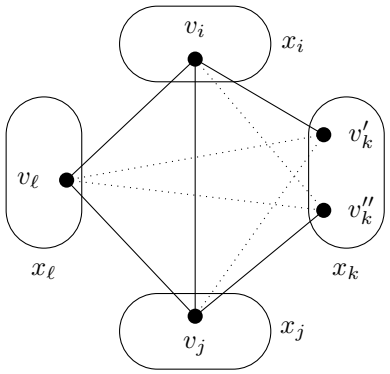

Fig. 2 A weakly broken-triangle (v'_k, v_i, v_j, v''_k) since $(v'_k, v_\ell), (v''_k, v_\ell) \notin Rel(C_{k\ell})$.

3 Flexible broken-triangles

A total absence of broken-triangles on a given variable in an arc-consistent CSP instance allows us to eliminate it without changing the satisfiability of the instance. In contrast, a total absence of weakly broken-triangles does not permit variable elimination.

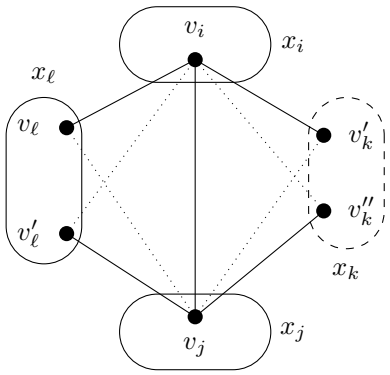

Fig. 3 The dashed variable x_k can be eliminated despite the presence of a broken-triangle.

Theoretically, we can define many examples of variables which can be eliminated despite the presence of certain broken-triangles while preserving satisfiability. As shown in the inconsistent CSP instance of Figure 3, there is a broken-triangle on v'_k and v''_k, but after eliminating x_k this CSP instance still remains inconsistent. So, the presence of some broken-triangle on a given variable does not preclude variable elimination while preserving satisfiability. For this, we introduce the *flexible broken-triangles*.

3.1 1-fBTP

Similar to m-wBTP, the m-fBTP is based on the concept of variable support. We begin by formally defining the simplest case (i.e. when $m = 1$).

Definition 8 A pair of values $v'_k, v''_k \in D(x_k)$ satisfies 1-**fBTP** if for each broken-triangle (v_i, v_j, v'_k, v''_k) with $v_i \in D(x_i)$, $v_j \in D(x_j)$, then there is at least one variable $x_\ell \in X \setminus \{x_i, x_j, x_k\}$ such that $\forall\ v_\ell \in D(x_\ell)$, if $(v_i, v_\ell) \in Rel(C_{i\ell})$ then $(v_j, v_\ell) \notin Rel(C_{j\ell})$. If this is the case, (v'_k, v_i, v_j, v''_k) is known as a **flexible broken-triangle** supported by the variable x_ℓ. A variable $x_k \in X$ satisfies 1-fBTP if each pair of values $v'_k, v''_k \in D(x_k)$ satisfies 1-fBTP.

In other words, each value in $D(x_\ell)$ cannot be compatible with both v_i and v_j at the same time. If there is no variable x_ℓ which satisfies the previous conditions, then the pair v'_k, v''_k does not satisfy 1-fBTP and (v'_k, v_i, v_j, v''_k) will be called *purely broken-triangle*.

We specify that the broken-triangle in Figure 3 is flexible because it is supported by the variable x_ℓ ($(v_i, v_\ell) \in Rel(C_{i\ell})$, $(v_j, v'_\ell) \in Rel(C_{j\ell})$, $(v_j, v_\ell) \notin Rel(C_{j\ell})$ and $(v_i, v'_\ell) \notin Rel(C_{j\ell})$).

We can intuitively deduce the following proposition:

Proposition 1 *In a binary CSP instance $I = (X, C)$, if a pair $v'_k, v''_k \in D(x_k)$ satisfies 1-fBTP, then it also satisfies 1-wBTP.*

Proof Straightforward. Indeed, to be 1-fBTP, we must have for each broken-triangle on v'_k and v''_k at least a support variable x_ℓ (for 1-fBTP) such that, each value v_ℓ in $D(x_\ell)$ is not compatible with both v_i and v_j at the same time. Thus, we do not have to check the compatibility of v_ℓ with v'_k and v''_k because x_ℓ is also a support variable for 1-wBTP. Finally, the pair $v'_k, v''_k \in D(x_k)$ also satisfies 1-wBTP. □

The converse is obiously false by means of Figure 2 where the pair (v'_k, v''_k) is 1-wBTP but is not 1-fBTP.

We immediately obtain the following result from Proposition 1 since m-wBTP allows value merging.

Corollary 1 *In a binary CSP instance $I = (X, C)$, merging a pair of values $v'_k, v''_k \in D(x_k)$ which satisfies 1-fBTP does not change the satisfiability of an instance.*

In the rest of this subsection, support variable will refer to fBTP.

It is known that if for a given variable x_k in an arc-consistent binary CSP instance I, the set of broken-triangles does not contain any pair of values v'_k, v''_k in $D(x_k)$ with two assignments to two other variables, then the variable x_k can be eliminated from I without modifying the satisfiability of I [8]. A similar result can also be shown for the variables satisfying 1-fBTP. To do it, we should prove the following lemma:

Lemma 1 *Given a variable x_k which satisfies 1-fBTP, after merging a pair of values $v''_k, v'''_k \in D(x_k)$ into a new value v'_k, no purely broken-triangle can appear on x_k.*

Proof We assume, for a contradiction, that after merging a pair of values v''_k, v'''_k of a variable x_k which satisfies 1-fBTP into a new value v'_k, we introduced a new purely broken-triangle (v_k, v_i, v_j, v'_k). This can be translated into the following relations:

- (1) $(v_i, v_j) \in Rel(C_{ij})$,
- (2) $(v_i, v_k) \in Rel(C_{ik})$,
- (3) $(v_j, v'_k) \in Rel(C_{jk})$,
- (4) $(v_j, v_k) \notin Rel(C_{jk})$ and
- (5) $(v_i, v'_k) \notin Rel(C_{ik})$.

By definition 5, we also have:
(5) $\Rightarrow$

- $(v_i, v''_k) \notin Rel(C_{ik})$ (a) and
- $(v_i, v'''_k) \notin Rel(C_{ik})$ (b).

(3) $\Rightarrow$

- either $(v_j, v''_k) \in Rel(C_{jk})$ (c)
- or $(v_j, v'''_k) \in Rel(C_{jk})$ (d).

(2), (1), (c), (a), and (4) $\Rightarrow$ a broken-triangle (v_i, v_j, v_k, v''_k) and (2), (1), (d), (b) and (4) $\Rightarrow$ a broken-triangle (v_k, v_i, v_j, v'''_k). In both cases, we had at least one broken-triangle before merging v''_k and v'''_k. So, there is at least one variable x_ℓ such that for each value $v_\ell \in D(x_\ell)$, if $(v_i, v_\ell) \in Rel(C_{i\ell})$ then $(v_j, v_\ell) \notin Rel(C_{j\ell})$. In this way, the variable x_ℓ also supports the broken-triangle (v_k, v_i, v_j, v'_k). Thus, (v_k, v_i, v_j, v'_k) is not a purely broken-triangle. But this contradicts our initial assumption. Therefore, merging two values v''_k, v'''_k in the domain of a variable x_k which satisfies 1-fBTP does not introduce a purely broken-triangle. □

We now establish the link with the variable elimination.

Theorem 1 *Given an arc-consistent CSP instance $I = (X, C)$, if a variable $x_k \in X$ satisfies 1-fBTP, then it can be eliminated from I while preserving satisfiability.*

Proof Given an arc-consistent CSP instance $I = (X, C)$ and a variable $x_k \in X$ which satisfies 1-fBTP. As value merging makes no empty domain, we will merge each pair of values in $D(x_k)$ until we obtain a unique value since (thanks to Lemma 1) merging a pair of values does not introduce a new purely broken-triangle on x_k. As I is arc-consistent, so any consistent assignment $\mathcal{A}$ to $X \backslash \{x_k\}$ can be easily extended to x_k because $D(x_k)$ contains a unique value and each value $\mathcal{A}[x_i]$ has a support in $D(x_k)$. So, the unique value in $D(x_k)$ is compatible with each value in $\mathcal{A}$. Thus, x_k can be eliminated without changing the satisfiability of I. □

Like the majority of BTP extensions, the principle of support variable introduced for flexible broken-triangles can be expanded to more than one variable. This will allows us to capture more variables that can be eliminated in binary CSP instances and to define a maximal variable-elimination condition.

3.2 m-fBTP

We now enlarge the definition of flexible broken-triangle property by using m support variables. These variables guarantee the incompatibility of at least one of their value with at least one of two values v_i and v_j of the broken trinagle on x_k. From a micro-structure viewpoint, these variables prevent the emergence of a new clique[2] which did not exist previously (as shown in figure 4).

We begin by formally defining m-fBTP.

[2] A complete subgraph where each pair of vertices are connected.

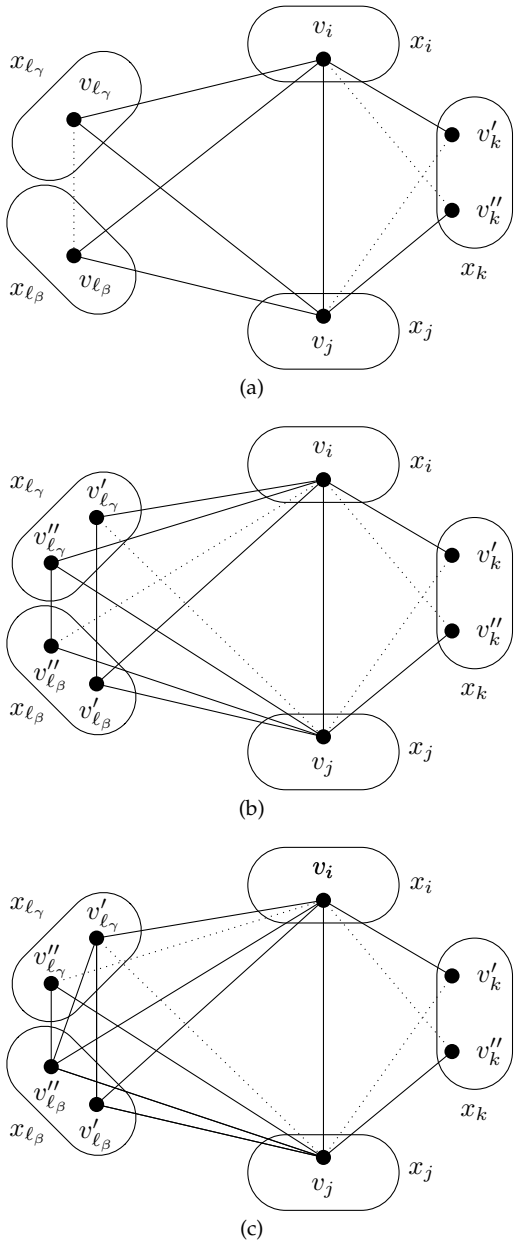

Fig. 4 Three different cases of two values v'_k and v''_k which satisfy 2-fBTP.

Definition 9 A pair of values $v'_k, v''_k \in D(x_k)$ satisfies m-**fBTP** where $m \leq n-3$ if for each broken-triangle (v'_k, v_i, v_j, v''_k) with $v_i \in D(x_i)$ and $v_j \in D(x_j)$, there is a set of $r \leq m$ **support variables** $\{x_{\ell_1}, \ldots, x_{\ell_r}\} \subseteq X \setminus \{x_i, x_j, x_k\}$ such that for all partial solution $(v_{\ell_1}, \ldots, v_{\ell_r}) \in D(x_{\ell_1}) \times \ldots \times D(x_{\ell_r})$, there is $\alpha \in \{1, \ldots, r\}$ such that if $(v_{\ell_\alpha}, v_i) \in Rel(C_{\ell_\alpha i})$, then $(v_{\ell_\alpha}, v_j) \notin Rel(C_{\ell_\alpha j})$. In this case, we say that

(v'_k, v_i, v_j, v''_k) is a flexible broken-triangle. A variable $x_k \in X$ satisfies m-fBTP if each pair of values $v'_k, v''_k \in D(x_k)$ satisfies m-fBTP.

As for 1-fBTP, if there is no set of m support variables which satisfies the previous conditions, then we will say that (v'_k, v_i, v_j, v''_k) is a *purely broken-triangle*.

Three different configurations of Definition 9 are given in Figure 4. In (a), there is no partial solution on the set of variables $\{x_{\ell_\beta}, x_{\ell_\gamma}\}$. Hence v'_k, v''_k in $D(x_k)$ clearly satisfies 2-fBTP. In (b), the pair of values v'_k, v''_k in $D(x_k)$ satisfies 2-fBTP because for the two partial solutions $(v'_{\ell_\beta}, v'_{\ell_\gamma})$ and $(v''_{\ell_\beta}, v''_{\ell_\gamma})$, we have $(v'_{\ell_\gamma}, v_j) \notin Rel(C_{\ell_\gamma j})$ and $(v''_{\ell_\beta}, v_i) \notin Rel(C_{\ell_\beta i})$. In (c), the pair of values v'_k, v''_k in $D(x_k)$ also satisfies 2-fBTP because for the two partial solutions $(v'_{\ell_\beta}, v'_{\ell_\gamma})$ and $(v''_{\ell_\beta}, v''_{\ell_\gamma})$, we have $(v'_{\ell_\gamma}, v_j) \notin Rel(C_{\ell_\gamma j})$ and $(v''_{\ell_\gamma}, v_i) \notin Rel(C_{\ell_\gamma i})$. Obviously, these three examples are unsolvable (inconsistent). But it is possible to make them consistent by adding new solutions whose values are completely disjoint with the values present in the examples. Unfortunately, this will make the figures too dense and difficult to understand.

Note that in Figure 4 (c), the variable x_{ℓ_γ} alone supports the broken-triangle (v'_k, v_i, v_j, v''_k), so we can deduce that x_{ℓ_γ} and x_{ℓ_β} together support it. In Figure 4 (a) and (b), x_{ℓ_γ} and x_{ℓ_β} together support the broken-triangle (v'_k, v_i, v_j, v''_k) none of them alone support it. From this, one can easily deduce the following result:

Proposition 2 *Given a binary CSP instance $I = (X, C)$, if a pair of values $v'_k, v''_k \in D(x_k)$ satisfies m-fBTP then it satisfies $(m+1)$-fBTP ($0 \leq m \leq n-4$).*

We now generalise Proposition 1 to any pair of values satisfying m-fBTP.

Proposition 3 *In a binary CSP instance $I = (X, C)$, $\forall m,\ 0 \leq m \leq n-4$, if a pair $v'_k, v''_k \in D(x_k)$ satisfies m-fBTP, then it also satisfies m-wBTP.*

Proof For each broken-triangle on v'_k, v''_k, there is a set of $r \leq m$ (with $0 \leq m \leq n-3$) support variables $\{x_{\ell_1}, \ldots, x_{\ell_r}\} \subseteq X \setminus \{x_i, x_j, x_k\}$ such that for all partial solution $(v_{\ell_1}, \ldots, v_{\ell_r}) \in D(x_{\ell_1}) \times \ldots \times D(x_{\ell_r})$, there is $\alpha \in \{1, \ldots, r\}$ such that if $(v_{\ell_\alpha}, v_i) \in Rel(C_{\ell_\alpha i})$, then $(v_{\ell_\alpha}, v_j) \notin Rel(C_{\ell_\alpha j})$. So each value v_{ℓ_α} in $D(x_{\ell_\alpha})$ cannot be compatible with v_i and v_j at the same time. Thus, there can be no partial solution $(v_{\ell_1}, \ldots, v_{\ell_r})$. As a result, the pair $v'_k, v''_k \in D(x_k)$ also satisfies m-wBTP. □

Corollary 2 *In a binary CSP instance $I = (X, C)$, merging a pair of values $v'_k, v''_k \in D(x_k)$ which satisfies m-fBTP does not change the satisfiability of I.*

If we denote by m-fBTP-merging the merging operation based on m-fBTP, we can deduce that 0-wBTP-merging [19] and 0-fBTP-merging correspond to BTP-merging defined in [6] since they are based on zero support variables. Since BTP-merging generalises both neighbourhood substitution [3] and virtual interchangeability [4] and m-fBTP-merging generalises BTP-merging for all $m \geq 0$, we immediately obtain the following results:

Corollary 3 *m-fBTP-merging generalises neighbourhood substitution and virtual interchangeability.*

Corollary 4 *m-fBTP-merging merges more values than BTP-merging and less than m-wBTP-merging.*

It is known that if a given variable x_k in an arc-consistent binary CSP instance I satisfies BTP then x_k can be eliminated without modifying the satisfiability of I [8]. A similar result can also be shown for the variables satisfying m-fBTP. To do it, we should prove the following lemma:

Lemma 2 *Given a variable x_k which satisfies m-fBTP, after merging a pair of values $v''_k, v'''_k \in D(x_k)$ into a new value v'_k, no purely broken-triangle can appear on x_k.*

Proof We assume, for a contradiction, that after merging a pair of values v''_k, v'''_k of a variable x_k which satisfies m-fBTP into a new value v'_k, we introduced a new purely broken-triangle (v_k, v_i, v_j, v'_k). So we have:

- (1) $(v_i, v_j) \in Rel(C_{ij})$,
- (2) $(v_i, v_k) \in Rel(C_{ik})$,
- (3) $(v_j, v'_k) \in Rel(C_{jk})$,
- (4) $(v_j, v_k) \notin Rel(C_{jk})$ and
- (5) $(v_i, v'_k) \notin Rel(C_{ik})$.

By definition 5, we obtain:

- $(v_i, v''_k) \notin Rel(C_{ik})$ (a),
- $(v_i, v'''_k) \notin Rel(C_{ik})$ (b), and
- either $(v_j, v''_k) \in Rel(C_{jk})$ (c) or $(v_j, v'''_k) \in Rel(C_{jk})$ (d).

(2), (1), (c), (a), and (4) $\Rightarrow$ a broken-triangle (v_k, v_i, v_j, v''_k) and (2), (1), (d), (b) and (4) $\Rightarrow$ a broken-triangle (v_k, v_i, v_j, v'''_k). In both cases, we had at least one broken-triangle before merging v''_k and v'''_k. So, there is a set of $r \leq m$ support variables $\{x_{\ell_1}, \ldots, x_{\ell_r}\} \subseteq X \setminus \{x_i, x_j, x_k\}$ such that for all partial solution $(v_{\ell_1}, \ldots, v_{\ell_r}) \in D(x_{\ell_1}) \times \ldots \times D(x_{\ell_r})$, there is $\alpha \in \{1, \ldots, r\}$ such that if $(v_{\ell_\alpha}, v_i) \in Rel(C_{\ell_\alpha i})$, then $(v_{\ell_\alpha}, v_j) \notin Rel(C_{\ell_\alpha j})$. In this way, the set of r support variables $\{x_{\ell_1}, \ldots, x_{\ell_r}\}$ also support the broken-triangle (v_k, v_i, v_j, v'_k). Thus, (v_k, v_i, v_j, v'_k) is not a purely broken-triangle. But this contradicts our initial assumption. Finally, merging two values v''_k, v'''_k in the domain of a variable x_k which satisfies m-fBTP does not introduce a purely broken-triangle. □

Lemma 2 cannot be extended to all pair of values which satisfies m-wBTP (and does not satisfy m-fBTP) [19]. Indeed, Figure 5(a) illustrates the case of a variable x_4 which satisfies 1-wBTP since the variable x_3 supports all the broken-triangles on x_4. Figure 5(b) is obtained after merging the values 2 and 1 into a new value 3. Hence, the variable x_3 no longer support the broken-triangle $(3, 2, 2, 0)$ (in bold) because the value $2 \in D(x_3)$ is compatible at the same time with 2 ($\in D(x_1)$), 2 ($\in D(x_2)$) and 3 ($\in D(x_4)$).

We now establish the link with variable elimination.

Theorem 2 *Given an arc-consistent CSP instance $I = (X, C)$, if a variable $x_k \in X$ satisfies m-fBTP, then it can be eliminated from I while preserving satisfiability.*

Proof Given an arc-consistent binary CSP instance $I = (X, C)$ and a variable $x_k \in X$ which satisfies m-fBTP. Since value merging makes no empty domain and does not affect the satisfiability of I (thanks to Corollary 2, we will merge each pair of values in $D(x_k)$ until obtaining a unique value since merging a pair of values does not introduce a new purely broken-triangle on x_k (thanks to Lemma 2).

As I is arc-consistent, so any consistent assignment $\mathcal{A}$ to $X \setminus \{x_k\}$ can be consistenly extended to x_k because $D(x_k)$ contains a unique value and each value $\mathcal{A}[x_i]$ has a support in $D(x_k)$. So, the unique value in $D(x_k)$ is compatible with each value in $\mathcal{A}$. Thus, x_k can be eliminated without changing the satisfiability of I. □

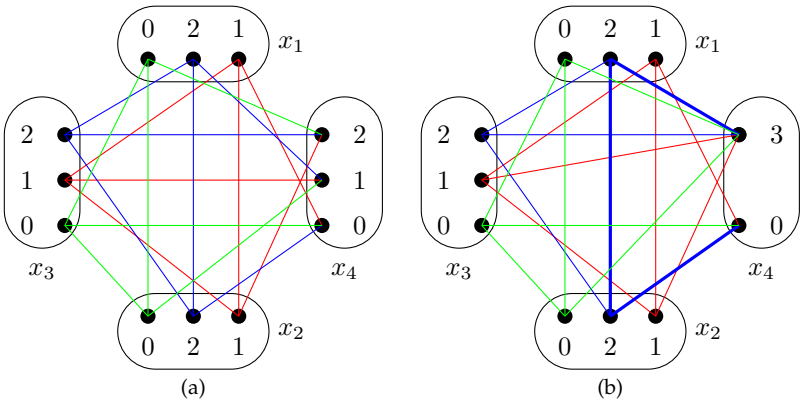

Fig. 5 (a) A variable x_4 which satisfies 1-wBTP in an arc-consistent CSP instance. (b) The CSP instance obtained from I after merging the values 1 and 2 into a new value 3.

4 A maximal variable-elimination condition

It has been proved that the variable which satisfies BTP can be eliminated while preserving satisfiability [8]. In section 3, we have shown that even if a variable does not satisfy BTP it can be eliminated without changing the satisfiability of the instance while this variable satisfies m-fBTP. Thus, in an obvious sense, satisfying BTP is not a maximal variable-elimination condition.

Definition 10 A variable-elimination condition is *maximal* if the elimination of any other variable not respecting the condition necessarily leads to a modification of the satisfiability of some instance.

In this section, we show that m-fBTP is a maximal variable-elimination condition when $m = n - 3$.

Theorem 3 *In an unsatisfiable binary CSP instance $I = (X, C)$, there is no variable not satisfying m-fBTP for $m = n-3$ and which can be eliminated while preserving satisfiability.*

Proof Considering an unsatisfiable binary CSP instance $I = (X, C)$ and a variable x_k which does not satisfy m-fBTP for $m = n - 3$. By the definition of m-fBTP, there is a broken-triangle (v'_k, v_i, v_j, v''_k), with $v_i \in D(x_i)$, $v_j \in D(x_j)$ and $v'_k, v''_k \in D(x_k)$. And there is $(v_{\ell_1}, \ldots, v_{\ell_m}) \in D(x_{\ell_1}) \times \ldots \times D(x_{\ell_m})$, where $\{x_{\ell_1}, \ldots, x_{\ell_m}\} = X \setminus \{x_i, x_j, x_k\}$, such that $(v_{\ell_1}, \ldots, v_{\ell_m})$ is a partial solution and for all $\alpha \in \{1, \ldots, m\}$ we have:

- $(v_{\ell_\alpha}, v_i) \in Rel(C_{\ell_\alpha i})$ and
- $(v_{\ell_\alpha}, v_j) \in Rel(C_{\ell_\alpha j})$

In terms of micro-structure we have a $(n-1)$-clique (a subset of $n-1$ vertices that induces a complete subgraph) that we denote Cl. We have a broken-triangle, and so:

- $(v_i, v''_k) \notin Rel(C_{ik})$,
- $(v_j, v'_k) \notin Rel(C_{jk})$,

- $(v_i, v'_k) \in Rel(C_{ik})$ and
- $(v_j, v''_k) \in Rel(C_{jk})$

After eliminating x_k, and by definition of elimination, the obtained instance I' has $(n-1)$ variables and its micro-structure contains the $(n-1)$-clique Cl. According to Property 2 in [25], Cl corresponds to a solution of I'. Thus, we introduced a solution which did not exist in the initial instance since $(v_i, v''_k) \notin Rel(C_{ik})$ and $(v_j, v'_k) \notin Rel(C_{jk})$. It follows that the elimination of variable which does not satisfy m-fBTP does not preserve satisfiability. □

We can now deduce the desired result.

Corollary 5 *$(n-3)$-fBTP is a maximal variable-elimination condition.*

5 m-fBTP: tractability and solving

In this section, we show the tractability of instances satisfying m-fBTP. Next, we prove that these instances will be efficiently solved by algorithms of the state-of-the-art like MAC (Maintaining Arc Consistency [20]) and RFL (Real Full Look-ahead [21]).

5.1 Tractability of m-fBTP instances

Contrary to k-BTP and m-wBTP which sometimes need a high level of consistency, we show that arc consistency is a decision procedure for m-fBTP. After defining m-fBTP for pair of values and variable, we now extend the definition to binary CSP instances.

Definition 11 A binary CSP instance I with a variable ordering $<$ satisfies m-fBTP relative to this order if for all variables x_k, each pair of values in $D(x_k)$ satisfies m-fBTP in the sub-instance of I on variables $x_i \leq x_k$ ($m \leq n-3$).

We now prove that m-fBTP is conservative[3] [11], m-fBTP holds even after enforcing any filtering consistency which only removes values from domains.

Lemma 3 *m-fBTP with respect to any fixed variable ordering is conservative.*

Proof It is clear that m-fBTP holds for a binary CSP instance thanks to the absence of some tuples. Obviously, removing values from the domain of any variable in a binary CSP instance cannot add new tuples. Thus, m-fBTP still holds. □

We now investigate the consequence of Lemma 3 on m-fBTP instances solving.

Theorem 4 *Arc consistency is a decision procedure for any binary CSP instance which satisfies m-fBTP ($1 \leq m \leq n-3$).*

[3] A class Γ of CSP instances is said to be conservative with respect to a filtering consistency ϕ if it is closed under ϕ, that is, if the instance obtained after the application of ϕ still belongs to Γ.

Proof Let $I = (X, C)$ be a binary CSP instance satisfying m-fBTP with respect to a variable ordering $<$. We begin by enforcing arc consistency. If this results to an empty domain, then obviously the obtained instance has no solution. Otherwise, thanks to Lemma 3, we know that the obtained instance will also satisfy m-fBTP. According to Theorem 2, we can proceed iteratively to eliminate the last variable with respect to $<$ until obtaining an instance with three variables x_1, x_2 and x_3. As I is becoming arc-consistent, so there is no empty domain. Hence, $D(x_1)$ (respectively $D(x_2)$) must contain at least a value v_1 (resp. v_2) such that $(v_1, v_2) \in Rel(C_{12})$ (1). We will suppose, for a contradiction, that the assignment $\mathcal{A} = (v_1, v_2)$ cannot be consistenly extended to x_3. For this, we assume that there is no $v_3 \in D(x_3)$ which is consistent with both v_1 and v_2. But, by arc consistency, we should have two values $v'_3, v''_3 \in D(x_3)$ such that

- $(v_1, v'_3) \in Rel(C_{13})$ (2) and
- $(v_2, v'_3) \in Rel(C_{23})$ (3)

Note that v'_3 and v''_3 must be different and $(v_1, v''_3) \notin Rel(C_{13})$ (4) and $(v_2, v'_3) \notin Rel(C_{23})$ (5) (otherwise we contradict our hypothesis).

In this way, (1), (2), (3), (4) and (5) form a purely broken-triangle on x_k which can be supported by no other variable. Indeed, by Definition 9, any variable x_ℓ must be different from $\{x_i, x_j, x_k\}$. Thus, this contradicts our assumption. Finally, $\mathcal{A}$ can be consistenly extended to x_3. □

The following theorem is a logical consequence of Corollary 5 and Theorem 4.

Theorem 5 *The class of binary CSP instances which satisfy $(n-3)$-fBTP defines the biggest tractable class resolved by variable elimination.*

As with m-wBTP, checking whether it is possible to compute, in polynomial time, a variable ordering for which a binary CSP instance satisfies m-fBTP still remains an open question.

5.2 Solving of m-fBTP instances by algorithms of the state-of-the-art

BTP and m-fBTP share many interesting properties. For example, the two tractable classes are conservative and solved by arc consistency. Hence, as BTP is solved in polynomial time by MAC, we will prove that MAC and RFL solve m-fBTP instances in polynomial time as well. Recall that both MAC and RFL guarantee arc consistency at each node of the search tree. The difference between them is that MAC is developing a binary search tree and RFL is developing a search tree with at most d branches at each node of the search tree. In addition, MAC does not necessarily choose the same variable at each level of the search tree (see [26] for more details on backtrack algorithms).

Theorem 6 *If a binary CSP instance I satisfies m-fBTP for an unknown variable ordering, then MAC and RFL solve I in polynomial time whatever the order of variables instanciation.*

Proof (Similar to the proof of Theorem 7.6. in [11]) Given a binary CSP instance I which satisfies m-fBTP, we deduce by Lemma 3 that any sub-instance of I, obtained after assigning a value v to a variable x, is also m-fBTP. By applying AC, the instance I either has a solution or has at least an empty domain. If there is an empty domain,

then I is unsatisfiable. Otherwise (if there is no empty domain), MAC or RFL will find at least one value in the domain (which is non-empty) of the next variable which will be compatible with all the values in the current assignment. In the worst case, MAC or RFL will check the compatibility of the d values (in the domain of the next variable) with the current assignment in each level of the search tree. This operation will take $O(nd)$. Thus, MAC and RFL will have a complexity $O(ned^3)$ with $O(ed^2)$ for enforcing arc consistency after assigning a value to the current variable. □

5.3 What about variable ordering?

To check whether a given binary CSP instance I satisfies BTP, Cooper et al. in [11] propose to construct a new CSP instance O^I which will be satisfiable when there exists a variable ordreing for I. O^I has the same set of variables as I but with different domains. Indeed, each variable has n values representing its possible positions in the ordering. For each broken-triangle (v'_k, v_i, v_j, v''_k) in I with $v_i \in D(x_i)$, $v_j \in D(x_j)$ and $v'_k, v''_k \in D(x_k)$, there is a constraint c in O^I over x_i, x_j and x_k which requires that $x_k < max(x_i, x_j)$. The instance O^I is max-closed [27] and so is tractable (see proof of Theorem 3.2. in [11] for more details).

For m-fBTP, we will proceed in a somewhat similar way. If there is a purely broken-triangle on a given variable x_k with respect to x_i and x_j, we add a new constraint c to O^I over x_i, x_j and x_k which requires that $x_k < max(x_i, x_j)$. And when there is a flexible broken-triangle on x_k with respect to x_i and x_j and which is supported by a variable x_ℓ, we have to add a less restrictive constraint which requires that `If` $x_k > max(x_i, x_j)$ `then` $x_\ell < max(x_i, x_j)$. This constraint will guarantee that x_ℓ is before x_k when x_k is after x_i and x_j in the variable ordering, as mentioned in Definition 11. In other words, if two variables x_i and x_j form a flexible broken-triangle on x_k and $x_k > max(x_i, x_j)$, the support variable x_ℓ must be before x_i or x_j, otherwise x_k does not satisfy 1-fBTP in the sub-instance of I on variables $x_i \leq x_k$.

Similarly, if the flexible broken-triangle on x_k with respect to x_i and x_j and which is supported by a set of support variables $\{x_{\ell_1}, x_{\ell_2}, \ldots, x_{\ell_m}\} \subseteq X \backslash \{x_i, x_j, x_k\}$, we have the following constraint which requires that `If` $x_k > max(x_i, x_j)$ `then` $x_{\ell_1} < max(x_i, x_j)$ and $x_{\ell_2} < max(x_i, x_j)$ and ... and $x_{\ell_m} < max(x_i, x_j)$.

Unfortunately, in this case, we do not know whether the instance O^I is tractable because their constraints are no longer max-closed. So, the question of the variable ordering for which a binary CSP instance satisfies m-fBTP still remains open.

Before concluding this section, we have to point out that, even if all the broken-triangles of a given binary CSP instance I are flexible, then I does not necessarily satisfy m-fBTP. Figure 6 shows the case of a binary CSP instance which does not satisfy 1-fBTP although all broken-triangles, listed below, are flexible.

- $(v'_i, v_\ell, v_k, v''_i)$, $(v'_i, v_\ell, v'_j, v''_i)$ and $(v'_i, v''_j, v_k, v''_i)$ on x_i, supported by x_j, x_k and x_ℓ, respectively.
- $(v'_j, v_\ell, v_k, v''_j)$, $(v'_j, v_\ell, v'_i, v''_j)$ and $(v'_j, v''_i, v_k, v''_j)$ on x_j, supported by x_i, x_k and x_ℓ, respectively.
- (v'_k, v_i, v_j, v''_k), $(v'_k, v_i, v_\ell, v''_k)$ and $(v'_k, v''_\ell, v_j, v''_k)$ on x_k, supported by x_ℓ, x_j and x_i, respectively.
- $(v'_\ell, v_i, v_j, v''_\ell)$, $(v'_\ell, v_i, v'_k, v''_\ell)$ and $(v'_\ell, v''_k, v_j, v''_\ell)$ on x_ℓ, supported by x_k, x_j and x_i, respectively.

These flexible broken-triangles impose the following constraints on the variable ordering:

- If $x_i > max(x_k, x_\ell)$ then $x_j < max(x_k, x_\ell)$
- If $x_i > max(x_\ell, x_j)$ then $x_k < max(x_\ell, x_j)$
- If $x_i > max(x_k, x_j)$ then $x_\ell < max(x_k, x_j)$
- If $x_j > max(x_k, x_\ell)$ then $x_i < max(x_k, x_\ell)$
- If $x_j > max(x_\ell, x_i)$ then $x_k < max(x_\ell, x_i)$
- If $x_j > max(x_k, x_i)$ then $x_\ell < max(x_k, x_i)$
- If $x_k > max(x_i, x_j)$ then $x_\ell < max(x_i, x_j)$
- If $x_k > max(x_\ell, x_i)$ then $x_j < max(x_\ell, x_i)$
- If $x_k > max(x_\ell, x_j)$ then $x_i < max(x_\ell, x_j)$
- If $x_\ell > max(x_i, x_j)$ then $x_k < max(x_i, x_j)$
- If $x_\ell > max(x_\ell, x_i)$ then $x_j < max(x_k, x_i)$
- If $x_\ell > max(x_\ell, x_j)$ then $x_i < max(x_k, x_j)$

Thus, it is impossible to find a variable ordering for which the instance satisfies 1-fBTP.

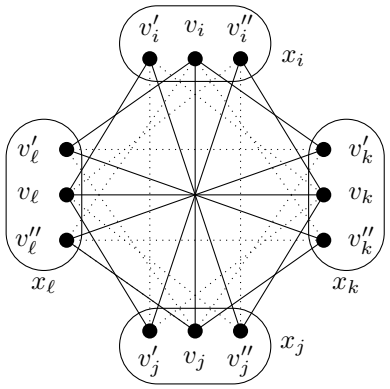

Fig. 6 A binary CSP instance which does not satisfy 1-fBTP whereas all broken-triangles are flexible.

6 m-fBTP vs some tractable classes based on BTP

BTP defines an important tractable class which has deserved to be studied and extended in many previous works (as mentioned in the introduction). Thus, it is natural to compare m-fBTP to some of these extensions such as DBTP [15], $\forall\exists$-BTP [13], BTP^{AC} [14], k-BTP [17] and WBTP [18]. For each class, we will show if it is a equality relation, inclusion or intersection.

6.1 DBTP

The *Dual Broken-Triangle Property* is an extension of BTP to non-binary CSPs by using the dual encoding [28]. Even for binary case, DBTP is different from BTP and authorises the presence of some broken-triangles which are forbidden by BTP.

Definition 12 (DBTP [29,15]) A binary CSP instance I satisfies DBTP with respect to a constraint ordering $\prec$ if and only if the dual of I satisfies BTP with respect to $\prec$.

Graphically, DBTP can be modelized as well as BTP, it suffices to replace each value in the micro-structure by a pair of compatible values in the micro-structure of the dual[4] [30]. For the simplicity of graphical representation, we will use v_iv_j to denote the compatible pair of values (v_i, v_j) in the figure of the micro-structure of the dual.

Theorem 7 *m-fBTP and DBTP are incomparable.*

Proof The binary CSP instance I of Figure 7 satisfies 1-fBTP with respect to the variable ordering $x_\ell < x_k < x_i < x_j$ despite the presence of the following flexible broken-triangles:

- (v_i, v_j, v'_k, v'_i) on x_i, supported by x_ℓ
- (v'_j, v'_i, v'_k, v_j) on x_j, supported by x_ℓ
- $(v''_k, v'_j, v'_i, v'_k)$ on x_k, supported by x_ℓ
- $(v_i, v_\ell, v'_j, v'_i)$ on x_i, supported by x_k
- (v'_j, v_ℓ, v_i, v_j) on x_j, supported by x_k

This imposes the following constraints on the variable ordering:

- $\texttt{If } x_i > max(x_k, x_j) \texttt{ then } x_\ell < max(x_k, x_j)$
- $\texttt{If } x_j > max(x_k, x_i) \texttt{ then } x_\ell < max(x_k, x_i)$
- $\texttt{If } x_k > max(x_i, x_j) \texttt{ then } x_\ell < max(x_i, x_j)$
- $\texttt{If } x_i > max(x_\ell, x_j) \texttt{ then } x_k < max(x_\ell, x_j)$
- $\texttt{If } x_j > max(x_\ell, x_i) \texttt{ then } x_k < max(x_\ell, x_i)$

At the same time, the micro-structure of the dual of I does not satisfy BTP on each of the following three constraints:

- $((v_j, v'_k), (v_i, v_j), (v_i, v''_k), (v'_j, v''_k))$ on C_{jk},
- $((v'_i, v'_j), (v'_j, v''_k), (v_i, v''_k), (v_i, v_j))$ on C_{ij} and
- $((v_i, v''_k), (v_i, v_j), (v_j, v'_k), (v'_i, v'_k))$ on C_{ik}.

So I does not satisfy DBTP.

On the other side, the binary CSP instance I of Figure 8 does not satisfy 1-fBTP whatever the variable ordering because of the following broken-triangles:

- (v_i, v''_k, v'_j, v'_i) on x_i,
- (v_j, v_i, v''_k, v'_j) on x_j and
- $(v'_k, v'_i, v'_j, v''_k)$ on x_k.

And there is no fourth variable that can support one of these broken-triangles. Futhermore, the micro-structure of the dual of I satisfies BTP with respect to the constraint ordering $C_{ij} < C_{ik} < C_{jk}$. So I satisfies DBTP.

Finally, we deduce that DBTP and 1-fBTP are incomparable. □ Obviously, the instance of figures 7 and 8 can be generalised to any $m > 1$ while maintaining the same logic.

[4] Given a binary CSP instance $I = (X, C)$, the *Micro-structure based on Dual* of I is the undirected graph (V, E) such that:

- $V = \{(C_i, t_i) : C_i \in C, t_i \in Rel(C_i)\}$,
- $E = \{\ \{(C_i, t_i), (C_j, t_j)\}\ \mid\ i \neq j, t_i[Scp(C_i) \cap Scp(C_j)] = t_j[Scp(C_i) \cap Scp(C_j)]\}$

where $t_k[Y]$ denotes the restriction of t_k to the variables in Y.

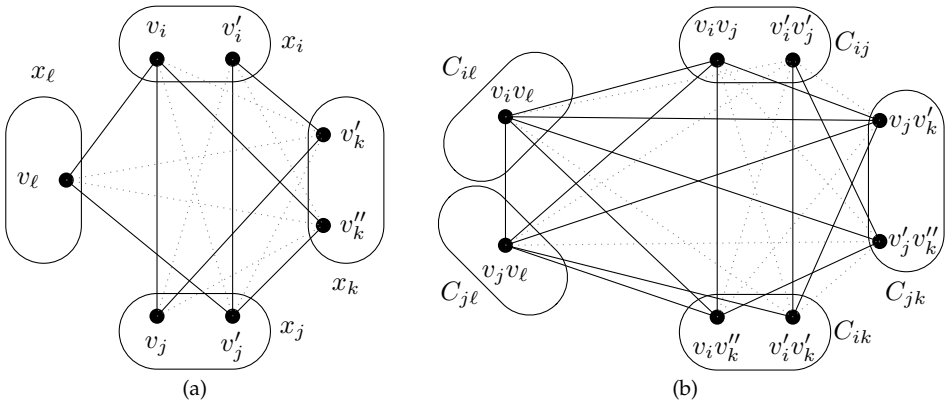

Fig. 7 (a) The micro-structure and (b) the micro-structure of the dual of a binary CSP instance I which satisfies 1-fBTP with respect to the order $x_\ell < x_k < x_i < x_j$ but does not DBTP whatever the constraint ordering.

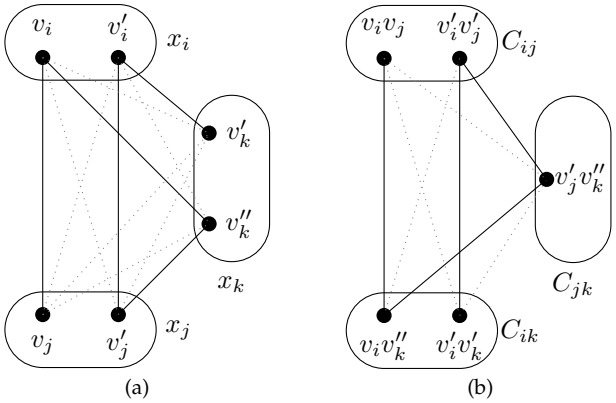

Fig. 8 (a) The micro-structure and (b) the micro-structure of the dual of a binary CSP instance which satisfies DBTP with respect to the order $C_{ij} < C_{ik} < C_{jk}$ but does not 1-fBTP whatever the variable ordering.

6.2 ∀∃-BTP

∀∃-BTP is a tractable class, introduced by Cooper in [13], and allows the existence of some broken-triangles.

Definition 13 (∀∃-BTP) A binary CSP instance I satisfies the property ∀∃-BTP with respect to a variable ordering $<$ if, and only if, for each pair of variables x_i, x_k such that $i < k$, $\forall v_i \in D(x_i)$, $\exists v_k \in D(x_k)$ such that $(v_i, v_k) \in Rel(C_{ik})$ and $\forall x_j$ with $j < k$ and $j \neq i$, and $\forall v_j \in D(x_j)$ and $\forall v'_k \in D(x_k)$, (v_i, v_j, v_k, v'_k) is not a broken-triangle on x_k with respect to x_i and x_j.

Theorem 8 defines the relationship between m-fBTP and ∀∃-BTP.

Theorem 8 *m-fBTP and ∀∃-BTP are incomparable.*

Proof Figure 9 shows a binary CSP instance which satisfies ∀∃-BTP but does not satisfy 1-fBTP because there is no fourth variable which can support the following broken-triangles:

- (v'_i, v_k, v_j, v''_i) on x_i,
- (v''_j, v_i, v_k, v'_j) on x_j and
- (v'_k, v_i, v_j, v''_k) on x_k.

For the binary CSP instance of Figure 8(a), we can observe that it satisfies 1-fBTP but does not satisfy ∀∃-BTP. □

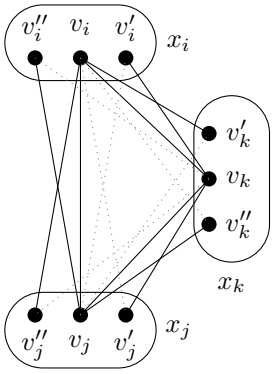

Fig. 9 A binary CSP instance which satisfies ∀∃-BTP whatever the variable ordering but does not satisfy 1-fBTP.

6.3 BTP^{AC}

The concept of *hidden* tractable class, obtained by applying some polynomial transformation, has been introduced in [14] to extend the power of already existing tractable classes. In this way, and for the case of BTP, the presence of several broken-triangles can be allowed, especially when a broken-triangle contains at least one inconsistent value. Here, we compare m-fBTP to the smallest hidden tractable class based on BTP, namely BTP^{AC} which is obtained by filtering by arc consistency.

Definition 14 (BTP^{AC} [14]) A binary CSP instance I satisfies BTP^{AC} if it satisfies BTP after enforcing arc consistency.

We now establish the link between m-fBTP and BTP^{AC}.

Theorem 9 *m-fBTP and BTP^{AC} are incomparable.*

Proof Despite the presence of the following broken-triangles:

- (v'_i, v''_k, v_j, v_i) on x_i,
- (v_j, v_i, v'_k, v'_j) on x_j and
- (v'_k, v_i, v_j, v''_k) on x_k.

Figure 10(a) shows an example of a binary CSP instance which satisfies BTP^{AC} (All broken-triangles will be removed after enforcing arc consistency) but does not 1-fBTP (because there is no support variable for all the broken-triangles).

- (v_i, v_j, v''_k, v'_i) on x_i which cannot be supported by x_ℓ because v_ℓ is compatible with both v_j and v''_k.
- (v_j, v_i, v'_k, v'_j) on x_j which cannot be supported by x_ℓ because v_ℓ is compatible with both v_i and v'_k.
- (v'_k, v_i, v_j, v''_k) on x_k which cannot be supported by x_ℓ because v_ℓ is compatible with both v_i and v_j.

Thus this instance does not satisfy 1-fBTP (and obviously m-fBTP). Figure 10(b)

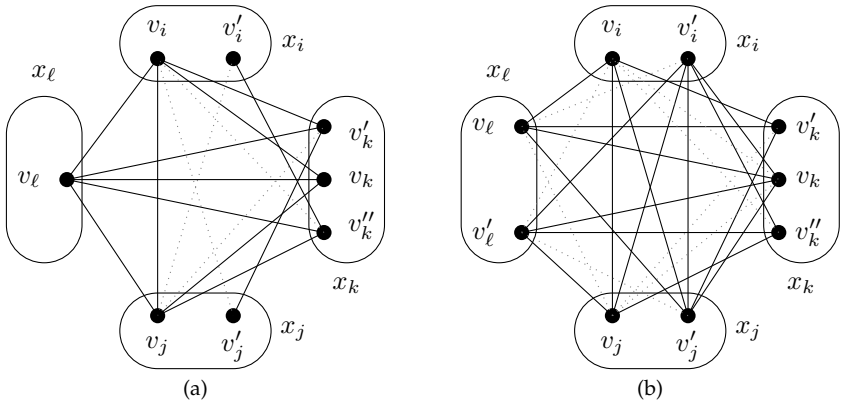

Fig. 10 (a) A binary CSP instance which satisfies BTP^{AC} but does not 1-fBTP. (b) A binary CSP instance which satisfies 1-fBTP but does not BTP^{AC}.

illustrates the case of a binary CSP instance which does not satisfy BTP^{AC} since each value in this instance is arc-consistent. On the other hand, this binary CSP instance satisfies 1-fBTP with respect to the variable ordering $x_\ell < x_i < x_j < x_k$ despite the presence of four broken-triangles (only one is flexible):

- $(v'_\ell, v_j, v_i, v_\ell)$ on x_ℓ,
- (v'_i, v_k, v_ℓ, v_i) on x_i,
- $(v_j, v'_\ell, v_k, v'_j)$ on x_j and
- (v'_k, v_i, v_j, v''_k) on x_k which is supported by x_ℓ

which imposes the following constraints on the variable ordering:

- $x_\ell < max(x_i, x_j)$,
- $x_i < max(x_\ell, x_k)$,
- $x_j < max(x_\ell, x_k)$,
- `If` $x_k > max(x_i, x_j)$ `then` $x_\ell < max(x_i, x_j)$.

Finally, BTP^{AC} and 1-fBTP are incomparable. □

6.4 k-BTP

k-BTP [17] is an extension of BTP which authorises some specific broken-triangles. Formally, it is defined as follows.

Definition 15 (k-BTP) A binary CSP instance I satisfies the k-BTP property for a given k ($2 \leq k < n$) relative to a variable order $<$ if, for all subsets of variables $x_{i_1}, x_{i_2}, \ldots, x_{i_{k+1}}$ such that $x_{i_1} < x_{i_2} < \cdots < x_{i_{k+1}}$, there is at least one pair of variables $(x_{i_j}, x_{i_{j'}})$ with $1 \leq j < j' \leq k$ such that there is no broken-triangle on x_{k+1} relative to x_{i_j} and $x_{i_{j'}}$.

The binary CSP instances which satisfy k-BTP do not define a tractable class if they are not strong strong k-consistent[5] [31].

Theorem 10 *[17] Given a binary CSP instance I such that there exists a constant k with $2 \leq k < n$ for which I satisfies both strong k-consistency and k-BTP with respect to the variable ordering $<$. Then I is consistent and a solution can be found in polynomial time.*

Theorem 11 *m-fBTP and k-BTP are incomparable.*

Proof Contrary to k-BTP which needs the strong k-consistency to be tractable, m-fBTP defines by itself a tractable class (it does not require any level of consistency). So if we consider a binary CSP instance I which satisfies m-fBTP but is not strong k-consistent, then even if I satisfies the property k-BTP, it will not be in the tractable class defined by k-BTP because it is not strong k-consistent.

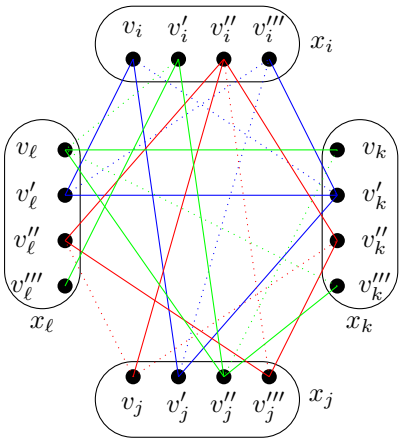

Fig. 11 A Binary CSP instance which satisfies the property 3-BTP with respect to every possible variable ordering but does not satisfy 1-fBTP, whatever the variable ordering.

Figure 11 illustrates the case of a binary CSP instance which does not satisfy k-BTP whatever the variable ordering because of the presence of the following flexible broken-triangles:

- $(v_i, v'_j, v'_k, v'''_i)$ on x_i which is supported by x_ℓ,
- $(v_i, v'_\ell, v'_k, v'''_i)$ on x_i which is supported by x_j,
- $(v_j, v''_i, v''_k, v'''_j)$ on x_j which is supported by x_ℓ,

[5] A binary CSP instance I 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 I satisfies strong k-consistency if it satisfies i-consistency for all i such that $1 < i \leq k$.

- $(v_j, v_i'', v_\ell'', v_j''')$ on x_j which is supported by x_k,
- $(v_k, v_\ell, v_j'', v_k''')$ on x_k which is supported by x_i and
- $(v_\ell, v_j'', v_i', v_\ell''')$ on x_ℓ which is supported by x_ℓ.

Each one of these flexible broken-triangles imposes the following constraints on the variable ordering:

- If $x_i > max(x_k, x_j)$ then $x_\ell < max(x_k, x_j)$,
- If $x_i > max(x_k, x_\ell)$ then $x_j < max(x_k, x_\ell)$,
- If $x_j > max(x_i, x_k)$ then $x_\ell < max(x_i, x_k)$,
- If $x_j > max(x_i, x_\ell)$ then $x_k < max(x_i, x_\ell)$,
- If $x_k > max(x_j, x_\ell)$ then $x_i < max(x_j, x_\ell)$ and
- If $x_\ell > max(x_i, x_j)$ then $x_k < max(x_i, x_j)$.

Thus, there is no possible variable ordering for which this binary CSP instance satisfies 1-fBTP. In contrast, this binary CSP instance satisfies the property 3-BTP. Obviously, for each tuple (v_p, v_q) in this binary CSP instance with $p, q \in \{i, j, k, \ell\}$ and $p \neq q$, we can add two values, one to x_g and the second to x_h with$g \neq h$ and $g, h \in \{i, j, k, \ell\} \setminus \{p, q\}$ such that the qudruplet (v_p, v_q, v_g, v_h) constitutes a partial solution. In this way, this binary CSP instance satisfies both 3-BTP and strong 3-consistency. □

6.5 WBTP

We finish this section with the recent tractable class called WBTP [18].

Definition 16 (WBTP) A binary CSP instance equipped with an order $<$ on its variables satisfies WBTP (Weak Broken-Triangle Property) if for each triple of variables $x_i < x_j < x_k$ and for all $v_i \in D(x_i)$, $v_j \in D(x_j)$ such that $(v_i, v_j) \in Rel(C_{ij})$, there is a variable $x_\ell < x_k$ such that when $v_\ell \in D(x_\ell)$ is compatible with v_i and v_j, then $\forall v_k \in D(x_k)$, if

- $(v_\ell, v_k) \in Rel(C_{\ell k})$

then

- $(v_i, v_k) \in Rel(C_{ik})$ and
- $(v_j, v_k) \in Rel(C_{jk})$

Theorem 12 *1-fBTP $\subsetneq$ WBTP.*

Proof Obviously because both WBTP and 1-fBTP use a unique support variable and their condition depends only from v_i and v_j. □

The converse of Theorem 12 is false by means of Figure 12. In fact, the binary CSP instance satisfies WBTP (anyone of x_{ℓ_β} and x_{ℓ_γ} supports all the broken-triangles) but does not satisfy 1-fBTP (more details will be given in the proof of Theorem 13).

Theorem 13 *2-fBTP and WBTP are incomparable.*

Proof Figure 12 shows a binary CSP instance which satisfies WBTP with respect to the variable ordering $x_{\ell_\beta} < x_{\ell_\gamma} < x_i < x_j < x_k$ but does not satisfy 2-fBTP. More precisely, there are three purely broken-triangles:

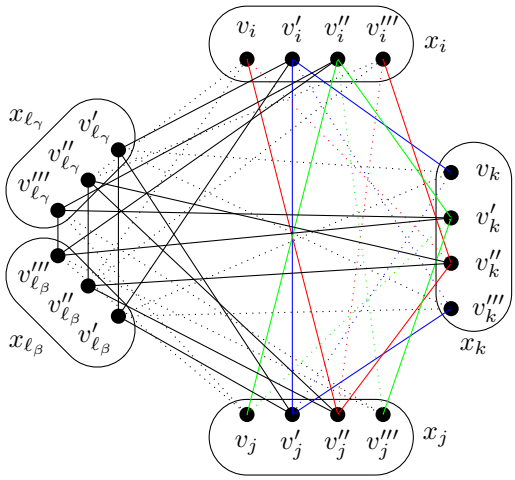

Fig. 12 A binary CSP instance which is WBTP but is not 2-fBTP.

- $(v_i, v''_j, v''_k, v'''_i)$ on x_i which cannot be supported by neither x_{ℓ_β} nor x_{ℓ_γ} (nor x_{ℓ_β} and x_{ℓ_γ} together) because v''_{ℓ_β} and v''_{ℓ_γ} are compatible with both v''_j and v''_k.
- $(v_j, v''_i, v'_k, v'''_j)$ on x_j which cannot be supported by neither x_{ℓ_β} nor x_{ℓ_γ} (nor x_{ℓ_β} and x_{ℓ_γ} together) because v'''_{ℓ_β} and v'''_{ℓ_γ} are compatible with both v''_i and v'_k.
- $(v_k, v'_i, v'_j, v'''_k)$ on x_k which cannot be supported by neither x_{ℓ_β} nor x_{ℓ_γ} (nor x_{ℓ_β} and x_{ℓ_γ} together) because v'_{ℓ_β} and v'_{ℓ_γ} are compatible with both v'_i and v'_j.

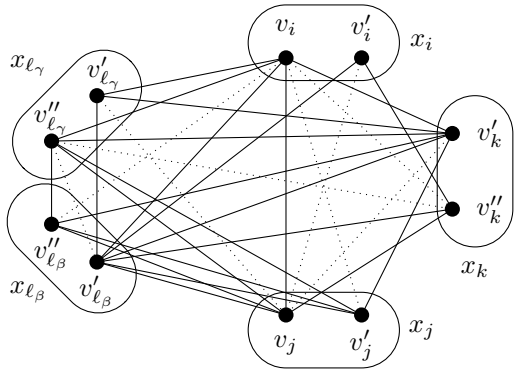

Fig. 13 A binary CSP instance which is 2-fBTP but is not WBTP.

Figure 13 illustrates the case of a binary CSP instance which does not satisfy WBTP but satisfies 2-fBTP with respect to the variable ordering $x_{\ell_\beta} < x_{\ell_\gamma} < x_i < x_j < x_k$ despite the presence of the following broken-triangles:

- (v_i, v_j, v''_k, v'_i) on x_i which is supported by x_{ℓ_β} and x_{ℓ_γ} together.
- (v_j, v_i, v'_k, v'_j) on x_j which is supported by x_{ℓ_β} and x_{ℓ_γ} together.
- (v'_k, v_i, v_j, v''_k) on x_k which is supported by x_{ℓ_β} and x_{ℓ_γ} together.

- $(v'_k, v''_{\ell_\gamma}, v_j, v''_k)$ on x_k which is supported by x_i.
- $(v'_{\ell_\gamma}, v'_{\ell_\beta}, v_j, v''_{\ell_\gamma})$ on x_{ℓ_γ} which cannot be supported by neither x_i nor x_k (nor x_i and x_k together).
- $(v'_{\ell_\beta}, v_i, v''_{\ell_\gamma}, v''_{\ell_\beta})$ on x_{ℓ_β} which cannot be supported by neither x_j nor x_k (nor x_j and x_k together).

They will impose the following constraints on the variable ordering:

- If $x_i > max(x_j, x_k)$ then $x_{\ell_\beta} < max(x_j, x_k)$ and $x_{\ell_\gamma} < max(x_j, x_k)$.
- If $x_j > max(x_i, x_k)$ then $x_{\ell_\beta} < max(x_i, x_k)$ and $x_{\ell_\gamma} < max(x_i, x_k)$.
- If $x_k > max(x_i, x_j)$ then $x_{\ell_\beta} < max(x_i, x_j)$ and $x_{\ell_\gamma} < max(x_i, x_j)$.
- If $x_k > max(x_j, x_{\ell_\beta})$ then $x_i < max(x_j, x_{\ell_\beta})$.
- $x_{\ell_\beta} < max(x_i, x_{\ell_\gamma})$.
- $x_{\ell_\gamma} < max(x_j, x_{\ell_\beta})$.

By considering the variable ordering $x_{\ell_\beta} < x_{\ell_\gamma} < x_i < x_j < x_k$, this binary CSP instance satisfies 2-fBTP. □

In the same way, we can show the following result:

Theorem 14 *for $m > 1$, m-fBTP and WBTP are incomparable.*

Figure 14 summarizes some relationship between tractable classes based on BTP. An arrow from c_1 to c_2 (resp. a dashed line between c_1 and c_2) means that $c_1 \subsetneq c_2$ (resp. c_1 and c_2 are incomparable). Obviously, a two-way arrow indicates equality.

Some of these relationships have been proved in this paper and some others in [14,32]. For several hidden tractable classes, more details about filtering by consistency can be found in [33,34].

7 Experimental trials

To test the existence of the property 1-fBTP, we we carried out an experimental study on all the binary benchmark instances of the 2008 international CSP solver competition[6], namely the 3,795 binary CSP instances. Our algorithm is written in C++ within our own CSP library. The experiments were performed on 8 Dell PowerEdge M820 blade servers with two processors (Intel Xeon E5-2609 v2 2.5 GHz and 32 GB of memory) under Linux Ubuntu 14.04.

Before applying our algorithm, we point out that we made each instance arc-consistent. As described in Subsection 5.3, we associate a non-binary CSP instance O to each benchmark. After that, we check, for each variable x_k, if there exists a broken-triangle on each pair of values $v'_k, v''_k \in D(x_k)$. Once a broken-triangle on v'_k, v''_k is found, we search over the other $n-3$ variables to see if there exists a variable x_ℓ which supports this broken-triangle. If we find one, we add a constraint which requires If $x_k > max(x_i, x_j)$ then $x_\ell < max(x_i, x_j)$. Otherwise, the broken-triangle on x_k, we add a new constraint c to O over x_i, x_j and x_k which requires that $x_k < max(x_i, x_j)$. Finally, we use MAC to check the satisfiability of the orginal instance. The previous result is tantamount to saying whether the original instance satisfies 1-fBTP.

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

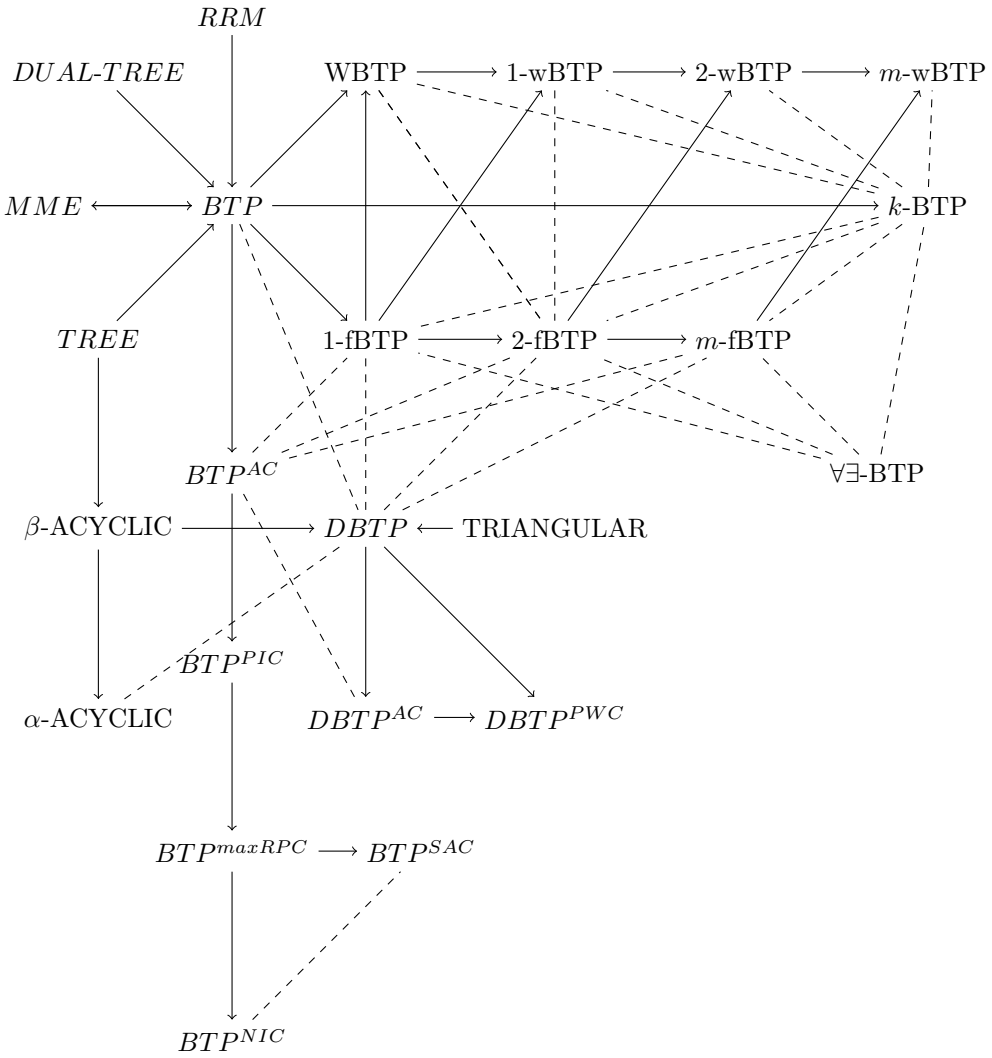

Fig. 14 Relationship between tractable properties based on BTP.

We obtained results for 3,260 instances. Among them, 280 instances satisfies 1-fBTP, including 46 consistent instances. All these instances also satisfy BTP after enforcing arc-consistency and solving by MAC and RFL without backtrack (more details are given in [14]).

8 Conclusion

BTP relies on absence of broken-triangle to define an important tractable class and to allow reducing search space size through value merging or variable elimination. Recently, many new weaker versions of BTP, which authorise the presence of some

broken-triangle like k-BTP, WBTP and m-wBTP, have been studied but none of them define tractable class and permit variable elimination and value merging simultaneously. Moreover, much of these versions, except WBTP, require a high level of consistency.

In this paper, we have proposed a new light version of BTP, called m-fBTP for flexible broken-triangle property. m-fBTP is based on support variable concept and permits to cover some imperfections of previous versions. More precisely, it allows value merging, represents a maximal variable-elimination condition and also defines a hybrid tractable class solved by arc consistency. m-fBTP is incomparable with the patterns described in [35] and which characterise tractable classes for CSPs defined by partially-ordered forbidden patterns and solved by arc consistency.

It would be interesting to generalise this family of definitions to non-binary CSPs. More generally, we have to study a new extension of BTP that is based on only three variables and which preserves its interesting characteristics.

References

1. U. Montanari, Networks of Constraints: Fundamental Properties and Applications to Picture Processing, Artificial Intelligence 7 (1974) 95–132.
2. A. K. Mackworth, Consistency in Networks of Relations, Artificial Intelligence 8 (1977) 99–118.
3. E. C. Freuder, Eliminating interchangeable values in constraint satisfaction problems, in: Proceedings of the 9th National Conference on Artificial Intelligence, 1991, Volume 1., 1991, pp. 227–233.
4. C. Likitvivatanavong, R. H. C. Yap, Many-to-many interchangeable sets of values in csps, in: Proceedings of SAC, 2013, pp. 86–91.
5. M. C. Cooper, A. El Mouelhi, C. Terrioux, B. Zanuttini, On Broken Triangles, in: Principles and Practice of Constraint Programming - 20th International Conference, CP 2014. Proceedings, 2014, pp. 9–24.
6. M. C. Cooper, A. Duchein, A. El Mouelhi, G. Escamocher, C. Terrioux, B. Zanuttini, Broken triangles: From value merging to a tractable class of general-arity constraint satisfaction problems, Artificial Intelligence 234 (2016) 196 – 218.
7. D. A. Cohen, M. C. Cooper, G. Escamocher, S. Zivny, Variable Elimination in Binary CSP via Forbidden Patterns, in: IJCAI 2013, Proceedings of the 23rd International Joint Conference on Artificial Intelligence, 2013, 2013, pp. 517–523.
8. D. A. Cohen, M. C. Cooper, G. Escamocher, S. Zivny, Variable and value elimination in binary constraint satisfaction via forbidden patterns, J. Comput. Syst. Sci. 81 (7) (2015) 1127–1143.
9. A. Dechter, R. Dechter, Removing redundancies in constraint networks, in: Proceedings of the 6th National Conference on Artificial Intelligence., 1987, pp. 105–109.
10. M. C. Cooper, P. Jeavons, A. Salamon, Hybrid tractable CSPs which generalize tree structure, in: Proceedings of ECAI, 2008, pp. 530–534.
11. M. C. Cooper, P. Jeavons, A. Salamon, Generalizing constraint satisfaction on trees: hybrid tractability and variable elimination, Artificial Intelligence 174 (2010) 570–584.
12. W. Naanaa, Unifying and extending hybrid tractable classes of csps, J. Exp. Theor. Artif. Intell. 25 (4) (2013) 407–424.
13. M. C. Cooper, Beyond consistency and substitutability, in: Principles and Practice of Constraint Programming - 20th International Conference, CP 2014. Proceedings, 2014, pp. 256–271.
14. A. El Mouelhi, P. Jégou, C. Terrioux, Hidden Tractable Classes: From Theory to Practice, in: 26th IEEE International Conference on Tools with Artificial Intelligence, ICTAI, 2014, 2014, pp. 437–445.
15. A. El Mouelhi, P. Jégou, C. Terrioux, A Hybrid Tractable Class for Non-Binary CSPs, Constraints 20 (4) (2015) 383–413.
16. P. Jégou, C. Terrioux, The extendable-triple property: A new CSP tractable class beyond BTP, in: Proceedings of AAAI, 2015, pp. 3746–3754.
17. M. C. Cooper, P. Jégou, C. Terrioux, A microstructure-based family of tractable classes for CSPs, in: Principles and Practice of Constraint Programming - 21st International Conference, CP, 2015, Proceedings, 2015, pp. 74–88.
18. W. Naanaa, Extending the broken triangle property tractable class of binary csps, in: Proceedings of the 9th Hellenic Conference on Artificial Intelligence, SETN, 2016, 2016, pp. 3:1–3:6.
19. M. C. Cooper, A. El Mouelhi, C. Terrioux, Extending broken triangles and enhanced value-merging, in: Principles and Practice of Constraint Programming - 22nd International Conference, CP 2016, Proceedings, 2016, pp. 173–188.
20. D. Sabin, E. C. Freuder, Contradicting Conventional Wisdom in Constraint Satisfaction, in: Proceedings of ECAI, 1994, pp. 125–129.
21. B. Nadel, Tree Search and Arc Consistency in Constraint-Satisfaction Algorithms, In *Search in Artificial Intelligence*, publisher = Springer-Verlag, year = 1988, pages = 287-342.
22. A. El Mouelhi, A btp-based family of variable elimination rules for binary csps, in: To appear in Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, 2017.
23. R. Dechter, J. Pearl, The Cycle-cutset method for Improving Search Performance in AI Applications, in: Proceedings of the third IEEE on Artificial Intelligence Applications, 1987, pp. 224–230.
24. F. Rossi, C. J. Petrie, V. Dhar, On the equivalence of constraint satisfaction problems, in: ECAI, 1990, pp. 550–556.
25. P. Jégou, Decomposition of Domains Based on the Micro-Structure of Finite Constraint Satisfaction Problems, in: Proceedings of the 11th National Conference on Artificial Intelligence, 1993., 1993, pp. 731–736.

26. A. El Mouelhi, P. Jégou, C. Terrioux, B. Zanuttini, Some new tractable classes of csps and their relations with backtracking algorithms, in: Integration of AI and OR Techniques in Constraint Programming for Combinatorial Optimization Problems, 10th International Conference, CPAIOR 2013, May 18-22, 2013. Proceedings, 2013, pp. 61–76.
27. P. Jeavons, M. Cooper, Tractable constraints on ordered domains, Artificial Intelligence 79(2) (1995) 327–339.
28. R. Dechter, J. Pearl, Tree-Clustering for Constraint Networks, Artificial Intelligence 38 (1989) 353–366.
29. A. El Mouelhi, P. Jégou, C. Terrioux, A Hybrid Tractable Class for Non-Binary CSPs, in: 2013 IEEE 25th International Conference on Tools with Artificial Intelligence, November 4-6, 2013, 2013, pp. 947–954.
30. A. El Mouelhi, P. Jégou, C. Terrioux, Microstructures for csps with constraints of arbitrary arity, in: Proceedings of the Tenth Symposium on Abstraction, Reformulation, and Approximation, SARA 2013, 11-12 July 2013, 2013.
31. E. C. Freuder, A Sufficient Condition for Backtrack-Free Search, JACM 29 (1) (1982) 24–32.
32. A. El Mouelhi, Classes polynomiales pour CSP : de la théorie à la pratique, Ph.D. thesis, Aix-Marseille Université (December 2014).
33. P. Jégou, On the consistency of general constraint-satisfaction problems, in: Proceedings of the 11th National Conference on Artificial Intelligence. July 11-15, 1993., 1993, pp. 114–119.
34. R. Debruyne, C. Bessière, Domain Filtering Consistencies, Journal of Artificial Intelligence Research 14 (2001) 205–230.
35. M. C. Cooper, S. Zivny, The power of arc consistency for CSPs defined by partially-ordered forbidden patterns, in: Proceedings of the 31st Annual ACM/IEEE Symposium on Logic in Computer Science, LICS '16, 2016, 2016, pp. 652–661.