On Broken Triangles

Martin Cooper, El Mouelhi, Cyril Terrioux, Bruno Zanuttini

On Broken Triangles*

Martin C. Cooper
IRIT
University of Toulouse 3
cooper@irit.fr

Achref El Mouelhi, Cyril Terrioux
Aix-Marseille University, LSIS
achref.elmouelhi@lsis.org
cyril.terrioux@lsis.org

Bruno Zanuttini
GREYC
Normandie University
bruno.zanuttini@unicaen.fr

Abstract

A binary CSP instance satisfying the broken-triangle property (BTP) can be solved in polynomial time. Unfortunately, in practice, few instances satisfy the BTP. We show that a local version of the BTP allows the merging of domain values in binary CSPs, thus providing a novel polynomial-time reduction operation. Experimental trials on benchmark instances demonstrate a significant decrease in instance size for certain classes of problems. We show that BTP-merging can be generalised to instances with constraints of arbitrary arity. A directional version of the general-arity BTP then allows us to extend the BTP tractable class previously defined only for binary CSP.

1 Introduction

At first sight one could assume that the discipline of constraint programming has come of age. On the one hand, efficient solvers are regularly used to solve real-world problems in diverse application domains while, on the other hand, a rich theory has been developed concerning, among other things, global constraints, tractable classes, reduction operations and symmetry. The research reported in this paper is part of a long-term project to bridge the gap between theory and practice.

Most research on tractable classes has been based on classes defined by placing restrictions either on the types of constraints or on the constraint hyper-graph whose vertices are the variables and whose hyper-edges are the constraint scopes. Another way of defining classes of binary CSP instances consists in imposing conditions on the microstructure, a graph whose vertices are the possible variable-value assignments with an edge linking each pair of compatible assignments [Jégou, 1993; Salamon and Jeavons, 2008]. If each vertex of the microstructure, corresponding to a variable-value assignment $\langle x, a\rangle$, is labelled by the variable x, then this so-called coloured microstructure retains all information from the original instance. The broken-triangle property (BTP) is a simple local condition on the coloured microstructure which defines a tractable class of binary CSP [Cooper *et al.*, 2010]. Inspired by the BTP, investigation of other forbidden patterns in the coloured microstructure has led to the discovery of new tractable classes [Cohen *et al.*, 2012; Cooper and Escamocher, 2015; Cooper and Živný, 2012; El Mouelhi *et al.*, 2015] as well as new reduction operations based on variable elimination [Cohen *et al.*, 2015].

For simplicity of presentation we use two different representations of constraint satisfaction problems. In the binary case, our notation is fairly standard, whereas in the general-arity case we use a notation close to the representation of SAT instances. This is for presentation only, though, and our algorithms do *not* need instances to be represented in this manner.

Definition 1 *A* binary CSP instance I *consists of*

- *a set* X *of* n *variables,*
- *a domain* $\mathcal{D}(x)$ *of values for each variable* $x \in X$*,*
- *a relation* $R_{xy} \subseteq \mathcal{D}(x) \times \mathcal{D}(y)$*, for each pair of distinct variables* $x, y \in X$*, which consists of the set of compatible pairs of values* (a, b) *for variables* (x, y)*.*

A partial solution *to* I *on* $Y = \{y_1, \ldots, y_r\} \subseteq X$ *is a set* $\{\langle y_1, a_1\rangle, \ldots, \langle y_r, a_r\rangle\}$ *such that* $\forall i, j \in [1, r]$, $(a_i, a_j) \in R_{y_i y_j}$*. A* solution *to* I *is a partial solution on* X*.*

For simplicity of presentation, Definition 1 assumes that there is exactly one constraint relation for each pair of variables. An instance I is *arc consistent* if for each pair of distinct variables $x, y \in X$, for each value $a \in \mathcal{D}(x)$, there is a value $b \in \mathcal{D}(y)$ such that $(a, b) \in R_{xy}$.

In our representation of general-arity CSP instances, we require the notion of *tuple* which is simply a set of variable-value assignments. For example, in the binary case, the tuple $\{\langle x, a\rangle, \langle y, b\rangle\}$ is *compatible* if $(a, b) \in R_{xy}$ and *incompatible* otherwise.

Definition 2 *A* (general-arity) CSP instance I *consists of*

- *a set* X *of* n *variables,*
- *a domain* $\mathcal{D}(x)$ *of values for each variable* $x \in X$*,*
- *a set NoGoods*(I) *consisting of incompatible tuples.*

A partial solution *to* I *on* $Y = \{y_1, \ldots, y_r\} \subseteq X$ *is a tuple* $t = \{\langle y_1, a_1\rangle, \ldots, \langle y_r, a_r\rangle\}$ *such that no subset of* t *belongs to NoGoods*(I)*. A* solution *is a partial solution on* X*.*

*supported by ANR Project ANR-10-BLAN-0210. Martin Cooper was also supported by EPSRC grant EP/L021226/1.

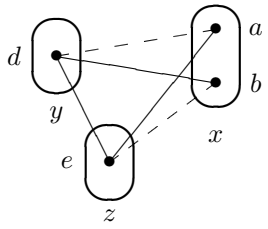

Figure 1: A broken triangle on two values a, b for a given variable x.

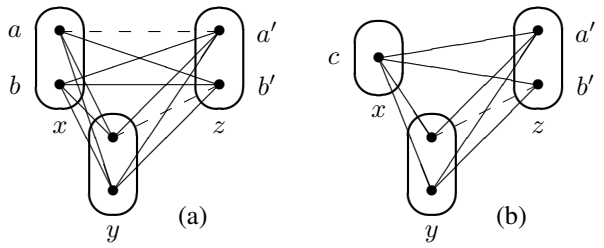

Figure 2: (a) A broken triangle exists on values a', b' at variable z. (b) After BTP-merging of values a and b in $\mathcal{D}(x)$, this broken triangle has disappeared.

2 Value merging in binary CSP

In this section we consider a method, based on the BTP, for reducing domain size while preserving satisfiability. Instead of eliminating a value, as in classic reduction operations such as arc consistency or neighbourhood substitution, we merge two values. We show that the absence of broken triangles on two values for a variable x in a binary CSP instance allows us to merge these two values in the domain of x while preserving satisfiability. This rule generalises the notion of virtual interchangeability [Likitvivatanavong and Yap, 2013] as well as neighbourhood substitution [Freuder, 1991].

It is known that if for a given variable x in an arc-consistent binary CSP instance I, the set of (in)compatibilities (known as a broken-triangle) shown in Figure 1 occurs for no two values $a, b \in \mathcal{D}(x)$ and no two assignments to two other variables, then the variable x can be eliminated from I without changing the satisfiability of I [Cooper *et al.*, 2010; Cohen *et al.*, 2015]. In figures, each bullet represents a variable-value assignment, assignments to the same variable are grouped together within the same oval and compatible (incompatible) pairs of assignments are linked by solid (broken) lines. Even when this variable-elimination rule cannot be applied, it may be the case that for a given pair of values $a, b \in \mathcal{D}(x)$, no broken triangle occurs. We will show that if this is the case, then we can perform a domain-reduction operation which consists in merging the values a and b.

Definition 3 Merging *values $a, b \in \mathcal{D}(x)$ in a binary CSP consists in replacing a, b in $\mathcal{D}(x)$ by a new value c which is compatible with all variable-value assignments compatible with at least one of the assignments $\langle x, a\rangle$ or $\langle x, b\rangle$. A* value-merging condition *is a polytime-computable property $P(x, a, b)$ of assignments $\langle x, a\rangle$, $\langle x, b\rangle$ in a binary CSP instance I such that when $P(x, a, b)$ holds, the instance I' obtained from I by merging $a, b \in \mathcal{D}(x)$ is satisfiable if and only if I is satisfiable.*

We now formally define the value-merging condition based on the BTP.

Definition 4 *A* broken triangle *on the pair of variable-value assignments $a, b \in \mathcal{D}(x)$ consists of a pair of assignments $d \in \mathcal{D}(y)$, $e \in \mathcal{D}(z)$ to distinct variables $y, z \in X \setminus \{x\}$ such that $(a, d) \notin R_{xy}$, $(b, d) \in R_{xy}$, $(a, e) \in R_{xz}$, $(b, e) \notin R_{xz}$ and $(d, e) \in R_{yz}$. The pair of values $a, b \in \mathcal{D}(x)$ is* BT-free *if there is no broken triangle on a, b.*

Proposition 5 *In a binary CSP instance, being BT-free is a value-merging condition. Furthermore, given a solution to the instance resulting from the merging of two values, we can find a solution to the original instance in linear time.*

Proof: Let I be the original instance and I' the new instance in which a,b have been merged into a new value c. Clearly, if I is satisfiable then so is I'. It suffices to show that if I' has a solution s which assigns c to x, then I has a solution. Let s_a, s_b be identical to s except that s_a assigns a to x and s_b assigns b to x. Suppose that neither s_a nor s_b are solutions to I. Then, there are variables $y, z \in X \setminus \{x\}$ such that $\langle a, s(y)\rangle \notin R_{xy}$ and $\langle b, s(z)\rangle \notin R_{xz}$. By definition of the merging of a, b to produce c, and since s is a solution to I' containing $\langle x, c\rangle$, we must have $(b, s(y)) \in R_{xy}$ and $(a, s(z)) \in R_{xz}$. Finally, $(s(y), s(z)) \in R_{yz}$ since s is a solution to I'. Hence, $\langle y, s(y)\rangle$, $\langle z, s(z)\rangle$, $\langle x, a\rangle$, $\langle x, b\rangle$ forms a broken-triangle, which contradicts our assumption. Hence, the absence of broken triangles on assignments $\langle x, a\rangle$, $\langle x, b\rangle$ allows us to merge these assignments while preserving satisfiability. Reconstructing a solution to I from a solution s to I' simply requires checking which of s_a or s_b is a solution to I. □ ■

The BTP-merging operation is not only satisfiability-preserving but, from Proposition 5, we know that we can also reconstruct a solution in polynomial time to the original instance I from a solution to an instance I^m to which we have applied a sequence of merging operations until convergence. Indeed, we have the following stronger result [Cooper *et al.*, 2014].

Proposition 6 *Let I be a binary CSP instance and suppose that we are given the set of all solutions to the instance I^m obtained after applying a sequence of BTP-merging operations. All N solutions to I can then be found in $O(Nn^2d)$ time.*

The weaker operation of neighbourhood substitution has the property that two different convergent sequences of eliminations by neighbourhood substitution necessarily produce isomorphic instances I_1^m, I_2^m [Cooper, 1997]. This is not the case for BTP-merging. Firstly, and perhaps rather surprisingly, BTP-merging can have as a side-effect to eliminate broken triangles. This is illustrated in the instance shown in Figure 2. The instance in Figure 2(a) contains a broken triangle on values a', b' for variable z, but after BTP-merging of values $a, b \in \mathcal{D}(x)$ into a new value c, as shown in Figure 2(b), there are no broken triangles in the instance. Secondly, BTP-merging of two values in $\mathcal{D}(x)$ can introduce a

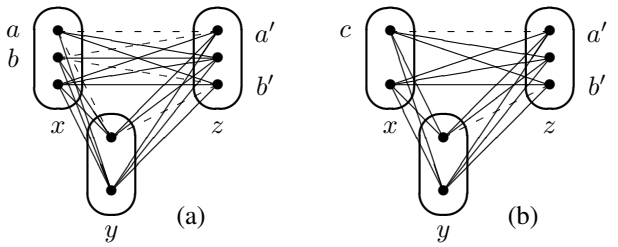

Figure 3: (a) This instance contains no broken triangle. (b) After BTP-merging of values a and b in $\mathcal{D}(x)$, a broken triangle has appeared on values $a', b' \in \mathcal{D}(z)$.

domain	N_{inst}	N_{val}	N_{del}	P_{del}
BH-4-13	6	7,334	3,201	44%
BH-4-4	10	674	322	48%
BH-4-7	20	2,102	883	42%
ehi-85	98	2,079	891	43%
ehi-90	100	2,205	945	43%
gr-col/school	8	4,473	104	2%
gr-col/sgb/book	26	1,887	534	28%
jobShop	45	6,033	388	6%
marc	1	6400	6,240	98%
os-taillard-4	30	2,932	1,820	62%
os-taillard-5	28	6,383	2,713	43%
rlfapGraphsMod	5	14,189	5,035	35%
rlfapScens	5	12,727	821	6%
rlfapScensMod	9	9,398	1,927	21%
others	1919	1,396	28	0.02%

Table 1: Results of experiments on CSP benchmark problems (N_{inst} = no. instances, N_{val} = no. values, N_{del} = no. values deleted, P_{del} = percentage deleted).

broken triangle on a variable $z \neq x$, as illustrated in Figure 3. The instance in Figure 3(a) contains no broken triangle, but after the BTP-merging of $a, b \in \mathcal{D}(x)$ into a new value c, a broken triangle has been created on values $a', b' \in \mathcal{D}(z)$. Indeed, it has been shown that finding an optimal sequence of BTP-merges is NP-hard [Cooper *et al.*, 2015].

3 Experimental trials

To test the utility of BTP-merging we performed extensive experimental trials on benchmark instances from the International CP Competition[1]. For each binary CSP instance, we performed BTP-mergings until convergence with a time-out of one hour. In total, we obtained results for 2,547 instances out of 3,811 benchmark instances within a time-out of one hour.

Table 1 gives a summary of the results of the experimental trials. We do not include those instances which are entirely solved by BTP-merging (such as all instances from the benchmark-domains `hanoi` and `domino`, or all instances from the `pigeons` benchmark-domain with a suffix `-ord`). We give details about those benchmark-domains where BTP-merging was most effective. All other benchmark-domains are grouped together in the last line of the table. The table shows the number of instances in the benchmark-domain, the average number of values (i.e. variable-value assignments) in the instances from this benchmark-domain, the average number of values deleted (i.e. the number of BTP-merging operations performed) and finally this average represented as a percentage of the average number of values.

We can see that for certain types of problem, BTP-merging is very effective, whereas for others (last line of the table) hardly any merging of values occurred. Runtime comparisons indicate that for BTP-merging to be useful in general-purpose solvers, we need to develop efficient algorithms to target those instances in which many merges are likely to occur [Cooper *et al.*, 2016].

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

4 BTP-merging: arbitrary-arity constraints

In the remainder of the paper, we assume that the constraints of a general-arity CSP instance I are given in the form described in Definition 2, i.e. as a set of incompatible tuples NoGoods(I), where a tuple is a set of variable-value assignments. For simplicity of presentation, we use the predicate Good(I, t) which is true iff the tuple t is a partial solution, i.e. t does not contain any pair of distinct assignments to the same variable and $\nexists t' \subseteq t$ such that $t' \in$ NoGoods(I). We first generalise the notion of broken triangle and merging to the general-arity case.

Definition 7 *A* general-arity broken triangle *(GABT) on values $a, b \in \mathcal{D}(x)$ consists of a pair of tuples t, u (containing no assignments to variable x) satisfying:*

1. *Good($I, t \cup u$) $\wedge$ Good($I, t \cup \{\langle x, a\rangle\}$) $\wedge$ Good($I, u \cup \{\langle x, b\rangle\}$)*
2. *$t \cup \{\langle x, b\rangle\},\ u \cup \{\langle x, a\rangle\} \in$ NoGoods(I)*

The pair of values $a, b \in \mathcal{D}(x)$ is GABT-free *if there is no broken triangle on a, b.*

Deciding whether a pair a, b is GABT-free is polytime for constraints given in extension (as the set of satisfying assignments) as well as for those given by nogoods (the set of assignments violating the constraint).

Definition 8 Merging *values $a, b \in \mathcal{D}(x)$ in a general-arity CSP instance I consists in replacing a, b in $\mathcal{D}(x)$ by a new value c which is compatible with all variable-value assignments compatible with at least one of the assignments $\langle x, a\rangle$ or $\langle x, b\rangle$, thus producing an instance I' with the new set of nogoods defined as follows:* NoGoods(I') $=$

$$\begin{aligned} & \{t \in \text{NoGoods}(I) \mid \langle x, a\rangle, \langle x, b\rangle \notin t\} \\ \cup\ & \{t \cup \{\langle x, c\rangle\} \mid t \cup \{\langle x, a\rangle\} \in \text{NoGoods}(I)\ \wedge \\ & \quad \exists t' \in \text{NoGoods}(I) \text{ s.t. } t' \subseteq t \cup \{\langle x, b\rangle\}\} \\ \cup\ & \{t \cup \{\langle x, c\rangle\} \mid t \cup \{\langle x, b\rangle\} \in \text{NoGoods}(I)\ \wedge \\ & \quad \exists t' \in \text{NoGoods}(I) \text{ s.t. } t' \subseteq t \cup \{\langle x, a\rangle\}\} \end{aligned}$$

A value-merging condition *is a polytime-computable property $P(x, a, b)$ of assignments $\langle x, a\rangle$, $\langle x, b\rangle$ in a CSP instance I such that when $P(x, a, b)$ holds, the instance I' is satisfiable if and only if I is satisfiable.*

This merging operation can be performed in polynomial time whether constraints are represented positively in extension or negatively as nogoods. As in the binary case, absence of general-arity broken triangles allows merging [Cooper *et al.*, 2014].

Proposition 9 *In a general-arity CSP instance I, being GABT-free is a value-merging condition. Furthermore, given a solution to the instance resulting from the merging of two values, we can find a solution to I in linear time.*

5 A tractable class of general-arity CSP

In binary CSP, the broken-triangle property defines an interesting tractable class when broken-triangles are forbidden according to a given variable ordering. Unfortunately, the original definition of BTP was limited to binary CSPs [Cooper *et al.*, 2010]. Section 4 described a general-arity version of the broken-triangle property whose absence on two values allows these values to be merged while preserving satisfiability. An obvious question is whether GABT-freeness can be adapted to define a tractable class. We will see that this is possible for a fixed variable ordering, but not if the ordering is unknown.

Definition 7 defined a general-arity broken triangle (GABT). What happens if we forbid GABTs according to a given variable ordering? Absence of GABTs on two values a, b for the last variable x in the variable ordering allows us to merge a and b while preserving satisfiability. It is possible to show that if GABTs are absent on all pairs of values for x, then we can merge all values in the domain $D(x)$ of x to produce a singleton domain. This is because merging a and b, to produce a merged value c, cannot introduce a GABT on c, d for any other value $d \in \mathcal{D}(x)$. Once the domain $D(x)$ becomes a singleton $\{a\}$, we can clearly eliminate x from the instance, by deleting $\langle x, a\rangle$ from all nogoods, without changing its satisfiability. It is at this moment that GABTs may be introduced on other variables, meaning that forbidding GABTs according to a variable ordering does not define a tractable class.

Nevertheless, strengthening the general-arity BTP allows us to avoid this problem. The resulting directional general-arity version of BTP (for a known variable ordering) then defines a tractable class which includes the binary BTP tractable class as a special case.

We suppose given a total ordering $<$ of the variables of a CSP instance I. We write $t^{<x}$ to represent the subset of the tuple t consisting of assignments to variables occurring before x in the order $<$, and $Vars(t)$ to denote the set of all variables assigned by t.

Definition 10 *A* directional general-arity (DGA) broken triangle *on assignments a, b to variable x in a CSP instance I is a pair of tuples t, u (containing no assignments to variable x) satisfying the following conditions:*

1. *$t^{<x}$ and $u^{<x}$ are non-empty*
2. *Good($I, t^{<x} \cup u^{<x}$) $\wedge$ Good($I, t^{<x} \cup \{\langle x, a\rangle\}$) $\wedge$ Good($I, u^{<x} \cup \{\langle x, b\rangle\}$)*
3. *$\exists t'$ s.t. $Vars(t') = Vars(t) \wedge (t')^{<x} = t^{<x} \wedge t' \cup \{\langle x, a\rangle\} \notin$ NoGoods(I)*
4. *$\exists u'$ s.t. $Vars(u') = Vars(u) \wedge (u')^{<x} = u^{<x} \wedge u' \cup \{\langle x, b\rangle\} \notin$ NoGoods(I)*
5. *$t \cup \{\langle x, b\rangle\},\ u \cup \{\langle x, a\rangle\} \in$ NoGoods(I)*

I satisfies the directional general-arity broken-triangle property (DGABTP) *according to the variable ordering $<$ if no directional general-arity broken triangle occurs on any pair of values a, b for any variable x.*

Any instance I satisfying the DGABTP can be solved in polynomial time by repeatedly alternating the following two operations: (i) merge all values in the last remaining variable (according to the order $<$); (ii) eliminate this variable when its domain becomes a singleton. Both operations preserve satisfiability and neither of them can introduce DGA broken triangles [Cooper *et al.*, 2014]. Moreover, the DGABTP can be tested in polynomial time for a given order.

Theorem 11 *A CSP instance I satisfying the DGABTP on a given variable ordering can be solved in polynomial time.*

An important question is the tractability of testing the existence of a variable ordering for which a given instance satisfies the DGABTP. Although this is polynomial-time for binary CSPs [Cooper *et al.*, 2010], it turns out to be NP-hard for general-arity CSPs [Cooper *et al.*, 2014].

Theorem 12 *Testing the existence of a variable ordering for which a CSP instance satisfies the DGABTP is NP-complete.*

6 Conclusion

This paper described a novel reduction operation for binary CSP, called BTP-merging, which is strictly stronger than neighbourhood substitution. Experimental trials have shown that in several benchmark-domains applying BTP-merging until convergence can significantly reduce the total number of variable-value assignments. We gave a general-arity version of BTP-merging and we then went on to define a general-arity version of the tractable class defined by the broken-triangle property for a known variable ordering.

References

[Cohen *et al.*, 2012] David A. Cohen, Martin C. Cooper, Páidí Creed, Dániel Marx, and András Z. Salamon. The tractability of CSP classes defined by forbidden patterns. *Journal of Artificial Intelligence Research*, 45, 47–78, 2012.

[Cohen *et al.*, 2015] David A. Cohen, Martin C. Cooper, Guillaume Escamocher and Stanislav Živný. Variable and value elimination in binary constraint satisfaction via forbidden patterns. *J. Comput. Syst. Sci.*, 81 (7), 1127–1143, 2015.

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

[Cooper *et al.*, 2015] Martin C. Cooper, Aymeric Duchein and Guillaume Escamocher. Broken Triangles Revisited. *Proceedings of CP*, *LNCS* 9255, 58–73, 2015.

[Cooper *et al.*, 2014] Martin C. Cooper, Achref El Mouelhi, Cyril Terrioux and Bruno Zanuttini. On Broken Triangles. *Proceedings of CP*, *LNCS* 8656, 9–24, 2014.

[Cooper *et al.*, 2016] Martin C. Cooper, Aymeric Duchein, Achref El Mouelhi, Guillaume Escamocher, Cyril Terrioux and Bruno Zanuttini. Broken Triangles: From Value Merging to a Tractable Class of General-Arity Constraint Satisfaction Problems. *Artificial Intelligence*, 234, 196–218, 2016.

[Cooper and Escamocher, 2015] Martin C. Cooper and Guillaume Escamocher. Characterising the complexity of constraint satisfaction problems defined by 2-constraint forbidden patterns. *Discrete Applied Mathematics*, 184, 89–113, 2015.

[Cooper *et al.*, 2010] Martin C. Cooper, Peter G. Jeavons, and András Z. Salamon. Generalizing constraint satisfaction on trees: Hybrid tractability and variable elimination. *Artificial Intelligence*, 174(9-10), 570–584, 2010.

[Cooper and Živný, 2012] Martin C. Cooper and Stanislav Živný. Tractable Triangles and Cross-Free Convexity in Discrete Optimisation, *Journal of Artificial Intelligence Research*, 44, 455–490, 2012.

[El Mouelhi *et al.*, 2015] Achref El Mouelhi, Philippe Jégou and Cyril Terrioux. A Hybrid Tractable Class for Non-Binary CSPs. *Constraints*, 20(4), 383–413, 2015.

[Jégou, 1993] Philippe Jégou. Decomposition of Domains Based on the Micro-Structure of Finite Constraint-Satisfaction Problems. *Proceedings of AAAI*, 731–736, 1993.

[Freuder, 1991] Eugene C. Freuder. Eliminating interchangeable values in constraint satisfaction problems. *Proceedings of AAAI*, 227–233, 1991.

[Likitvivatanavong and Yap, 2013] Chavalit Likitvivatanavong and Roland H.C. Yap. Eliminating redundancy in CSPs through merging and subsumption of domain values. *ACM SIGAPP Applied Computing Review*, 13(2), 2013.

[Salamon and Jeavons, 2008] András Z. Salamon and Peter G. Jeavons. Perfect Constraints Are Tractable. In *Proceedings of CP*, *LNCS* 5202, 524–528, 2008.