Conservative Dual Consistency

Stéphane Cardon, Christophe Lecoutre and Julien Vion

{cardon,lecoutre,vion}@cril.univ-artois.fr

Introduction

- Consistencies are properties of Constraint Networks (CNs) that can be exploited in order to make inferences and make CNs much easier to solve.
- We propose a new consistency called Dual Consistency (DC) and relate it to Path Consistency (PC).
- We show that Conservative DC (CDC, i.e. DC with only relations associated with the constraints of the network considered) is more powerful, in terms of filtering, than Conservative PC (CPC).
- Following the approach of Mac Gregor [1], we introduce an algorithm to establish (strong) CDC with a very low worst-case space complexity.
- The experiments we have conducted show that, on many series of CSP instances, CDC is largely faster than CPC (up to more than one order of magnitude).

Constraint Networks and Consistencies

Definition 1. A Constraint Network is a pair $(\mathscr{X}, \mathscr{C})$ where:

- $\mathscr{X}$ is a finite set of variables. Their associated domains $\mathrm{dom}(X)$, represents the set of values allowed for X.
- $\mathscr{C}$ is a finite set of constraints. Each constraint $C \in \mathscr{C}$ describes the set of allowed tuples $\mathrm{rel}(C)$ for variables $\mathrm{scp}(C)$.

We restrict our attention to binary networks and consider that the same scope cannot be shared by two distinct constraints.

Notations

- X_a: a pair (X, a) with $X \in \mathscr{X}$ and $a \in \mathrm{dom}(X)$
- n: the number of variables
- e: the number of constraints
- d: the largest domain size
- λ: the number of allowed tuples over all constraints of P
- K: the number of 3-cliques in P.
- D: the density of the binary CN ($e/\binom{n}{2}$)

The Constraint Satisfaction Problem (CSP) is the NP-complete task of determining whether a given constraint network is satisfiable, i.e. admits at least one assignment of values to all the variables that satisfies all constraints.

Consistencies are enforced to on a CN to identify and removing some inconsistent values or pairs of values. From now on, we will consider a binary constraint network $P = (\mathscr{X}, \mathscr{C})$.

Definition 2. A value X_a of P is arc-consistent (AC) iff $\forall C \in \mathscr{C} \mid X \in \mathrm{scp}(C), \exists (X_a, Y_b) \in \mathrm{rel}(C) \mid b \in \mathrm{dom}(Y)$.
A CN P is AC iff $\forall X_a | X \in (X) \wedge a \in \mathrm{dom}(X)$, X_a is AC.

Definition 3. A pair of values (X_a, Y_b) (with $X \neq Y$) is

- path-consistent (PC) iff $\forall Z \in \mathscr{X} \mid Z \notin \{X, Y\}$ $\exists \{C, C'\} \in \mathscr{C}^2 \mid \mathrm{scp}(C) = \{X, Z\} \wedge \mathrm{scp}(C') = \{Y, Z\} \wedge \exists c \in \mathrm{dom}(Z) \mid (X_a, Z_c) \in \mathrm{rel}(C) \wedge (Y_b, Z_c) \in \mathrm{rel}(C')$
- conservative path-consistent (CPC) iff either $\nexists C \in \mathscr{C} \mid \mathrm{scp}(C) = \{X, Y\}$ or (X_a, Y_b) is PC.

Definition 4. P is PC (resp. CPC) iff $\forall \{X_a, Y_b\} \mid \{X, Y\} \in \mathscr{X}^2 \wedge X \neq Y, \{X_a, Y_b\}$ is PC (resp. CPC).

For all considered consistencies ϕ and any CN P, there exists a greatest subnetwork of P which is ϕ-consistent, denoted by $\phi(P)$, and it is possible to compute it in polynomial time. For example, $\mathrm{AC}(P)$ is such that all values of P that are not arc-consistent have been removed. If any variable in $\phi(P)$ has an empty domain, P is unsatisfiable ($\phi(P) = \bot$). $P|_{X=a}$ denotes the network obtained from P by restricting the domain of X to the singleton $\{a\}$.

Definition 5. A pair (X_a, Y_b) of values of P s.t. $X \neq Y$ is:

- dual-consistent (DC) iff $Y_b \in \mathrm{AC}(P|_{X=a})$ and $X_a \in \mathrm{AC}(P|_{Y=b})$.
- conservative dual-consistent (CDC) iff either $\nexists C \in \mathscr{C} \mid \mathrm{scp}(C) = \{X, Y\}$ or (X_a, Y_b) is DC.

Definition 6. P is DC (resp. CDC) iff $\forall \{X_a, Y_b\} \mid \{X, Y\} \in \mathscr{X}^2 \wedge X \neq Y, \{X_a, Y_b\}$ is DC (resp. CDC).

Qualitative Study

Notation $\phi \succ \psi$: ϕ is strictly stronger than ψ (whenever ϕ holds on a CN P, ψ also holds on P and there exists at least one CN P such that ϕ holds on P but not ψ.)

Theorem 1. $\mathrm{DC} = \mathrm{PC}$

In the following figures, edges correspond to allowed tuples.

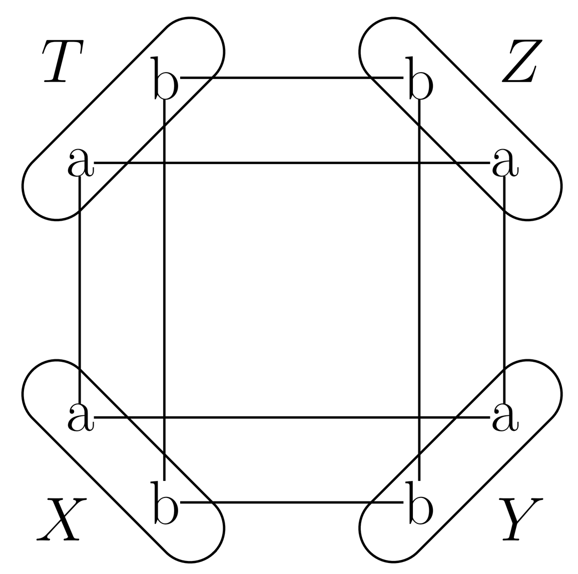

Figure 1: *A network (no constraint binds X with Z and Y with T) that is CDC but not PC. For example, (X_a, Z_b) is not PC.*

Theorem 2. $\mathrm{PC} \succ \mathrm{CDC}$.

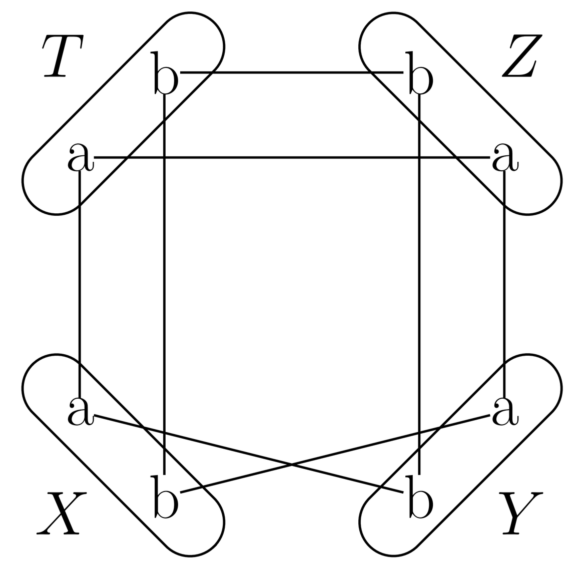

Figure 2: *A network (no constraint binds X with Z and Y with T) that is CPC but not CDC. For example, (X_a, T_a) is not CDC as $\mathrm{AC}(P|_{X=a}) = \bot$.*

Theorem 3. $\mathrm{CDC} \succ \mathrm{CPC}$.

Algorithm sCDC-1

The following algorithm enforces sCDC, i.e. ensures that the resulting network is AC and CDC.

Algorithm: sCDC-1($P = (\mathscr{X}, \mathscr{C})$: CN)

```
P ← AC(P, 𝒳);
marker ← X ← first(𝒳);
repeat
    if check(X) then
        P ← AC(P, {X});
        marker ← X;
    X ← next-modulo(X, 𝒳);
until X ≠ marker ;
```

Algorithm: check(P : CN, X : Variable) : Boolean

```
modified ← false
foreach a ∈ dom^P(X) do
    P' ← AC(P|_{X=a}, {X})
    if P' = ⊥ then
        remove a from dom^P(X)
        modified ← true
    else
        foreach C ∈ 𝒞 | X ∈ scp(C) do
            let Y be the second variable in scp(C)
            foreach b ∈ dom^P(Y) | b ∉ dom^{P'}(Y) do
                remove (X_a, Y_b) from rel^P(C)
                modified ← true
return modified
```

Theorem 4. The worst-case time complexity of sCDC-1 is $O(\lambda end^3)$ and its worst-case space complexity is $O(ed^2)$.

Note that $O(\lambda end^3) \subseteq O(e^2nd^5)$.

Corollary 1. *Applied to a sCDC network, the worst-case time complexity of sCDC-1 is $O(end^3)$.*

Corollary 2. *The best-case time complexity of sCDC-1 is $O(ed^2)$.*

Experiments

Experiments are performed on various set of instances from the second CSP solvers competition[2]. In the following, λ gives a measurement of the filtering done by the different algorithm: the smallest λ is, the smallest the resulting CN is (and thus, the filtering provided powerful is).

Experimental results on various series of instances

	AC3rm	SAC-SDS	sCPC8 / sCPC2001	sCDC-1
Langford (4 instances)				
cpu	0.22	0.46	4.02 / 4.94	0.52
λ	105,854	105,769	75,727	75,727
blackhole-4-13 (7 instances) $(K = 92,769 ; D = 20\%)$				
cpu	1.26	19.39	140.54 / –	46.91
λ	8,206,320	8,206,320	8,206,320	7,702,906
$\langle 40, 180, 84, 0.9 \rangle$ (20 instances) $(K = 12 ; D = 10\%)$				
cpu	0.71	10.57	2.28 / 2.02	17.42
λ	272,253	244,887	244,272	210,874
$\langle 40, 8, 753, 0.1 \rangle$ (20 instances) $(K = 8,860 ; D = 96\%)$				
cpu	0.16	0.21	0.62 / 0.69	0.20
λ	43,320	43,320	43,318	43,318
job-shop enddr1 (10 instances) $(K = 600 ; D = 21\%)$				
cpu	1.58	4.06	7.91 / 10.54	4.67
λ	2,937,697	2,937,697	2,937,697	2,930,391
RLFAP scens (11 instances)				
cpu	0.86	–	25.96 / –	3.47
λ	1,674,286	–	1,471,132	1,469,286

Experimental results on various single instances

	AC3rm	SAC-SDS	sCPC8 / sCPC2001	sCDC-1
driverlogw-09 $(K = 233,834 ; D = 8\%)$				
cpu	1.60	48.42	33.84 / 36.52	10.83
mem	14	87	59 / 155	23
λ	369,736	147,115	306,573	18,958
haystack-40 $(K = 395,200 ; D = 2\%)$				
cpu	9.64	–	580.48 / –	55.91
mem	19	–	209 / –	107
λ	48,670,518	–	48,670,518	48,670,518
knights-50-5 $(K = 10 ; D = 100\%)$				
cpu	12.38	34.43	1759 / –	21.49
mem	5	163	29 / –	19
λ	31,331,580	0	0	0
pigeons-50 $(K = 19,600 ; D = 100\%)$				
cpu	1.38	2.85	33.82 / 44.52	2.7
mem	2	12	9 / 636	5
λ	2,881,200	2,881,200	2,881,200	2,881,200
qcp-25-264-0 $(K = 43,670 ; D = 5\%)$				
cpu	2.28	6.08	8.15 / 10.49	2.08
mem	8	210	29 / 215	21
λ	77,234	77,234	76,937	76,937
qwh-25-235-0 $(K = 35,700 ; D = 4.5\%)$				
cpu	1.87	5.62	7.09 / 9.05	2.56
mem	7	183	26 / 173	19
λ	56,721	56,721	56,380	56,380
fapp01-200-4 $(K = 247 ; D = 0.5\%)$				
cpu	10.73	–	16.05 / 18.63	104.05
mem	15	–	22 / 254	17
λ	3,612,163	–	3,317,135	2,117,575
scen-11 $(K = 13,775 ; D = 1.7\%)$				
cpu	2.87	–	85.82 / 78.49	9.78
mem	5	–	22 / 426	16
λ	5,434,107	–	4,829,442	4,828,650

Impact of sCDC at preprocessing on MAC

$Instance$		MAC	$sCDC + MAC$
scen11-f8	cpu	8.08	14.31
	$nodes$	14,068	4,946
scen11-f5	cpu	259	225
	$nodes$	1,327K	680K
scen11-f3	cpu	2,338	1,725
	$nodes$	12M	5,863K
scen11-f2	cpu	7,521	5,872
	$nodes$	37M	21M
scen11-f1	cpu	17,409	13,136
	$nodes$	93M	55M

(cpu in seconds, mem in MiB)

Conclusion

We have introduced a new consistency called Dual Consistency (DC) and have focused on its conservative variant CDC. It has been shown in particular that CDC is a relation filtering consistency which is stronger than conservative PC (CPC), and enforcing strong CDC (i.e. enforcing both CDC and AC) can be done in a quite natural way (sCDC is also stronger than sCPC and easier to obtain). The experimental results obtained from a wide range of problems clearly show the practical interest of CPC, in particular on hard dense problems.

References

[1] J.J. McGregor. Relational consistency algorithms and their application in finding subgraph and graph isomorphisms. *Information Sciences*, 19:229–250, 1979.

[2] M. van Dongen, C. Lecoutre, O. Roussel, R. Szymanek, F. Hemery, C. Jefferson, and R. Wallace. Second International CSP Solvers Competition. http://cpai.ucc.ie/06/Competition.html, 2006.